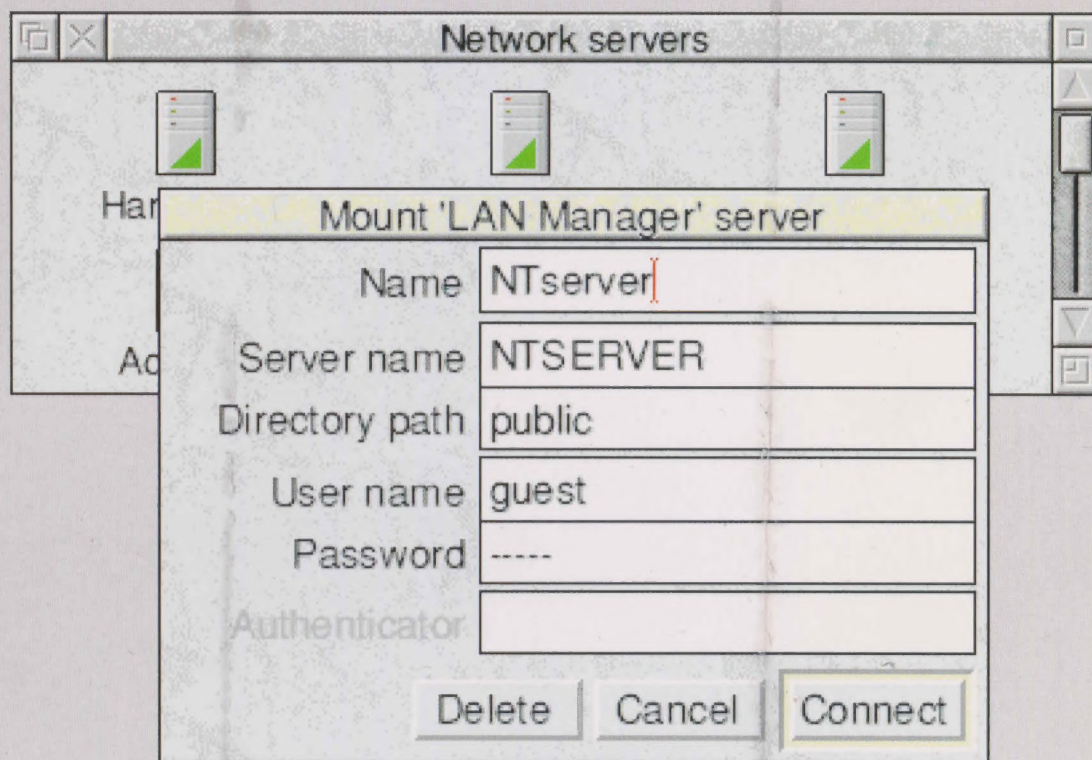




# Omniclient



## User Guide





ANT Ltd

# OmniClient User Guide

Issue 1, July 1995



© Copyright 1995 ANT Limited, Cambridge, England

!BootNet and !Internet kindly supplied by Acorn Computers Ltd.

© Acorn Computers Ltd 1995.

Issue 1, July 1995

Neither the whole nor any part of the information contained in, or the product described in, this manual may be adapted or reproduced in any material or electronic form without the prior written consent of the copyright holder. See Licence agreement on page 5 for full information about the product licence.

This product and its documentation are supplied on an as is basis and no warranty as to their suitability for any particular purpose is either made or implied. ANT Limited will not accept any claim for damages howsoever arising as a result of use or failure of this product. Your statutory rights are not affected.

ANT OmniClient is not intended for use in any medical appliance, device or system in which the failure of the product might reasonably be expected to result in personal injury.

ANT OmniClient was created by:

Project manager

Alex van Someren

Programmers

Nick Smith

Ian Harvey

David Fell

Dr NBS van Someren

Documentation

Carol Atack

Our thanks also to: Acorn Computers for supplying parts of this documentation.

OmniClient® is a registered trademark of ANT Limited. All other trade marks acknowledged.

ANT Limited, PO Box 300, Cambridge, England, CB1 3EG

Telephone: 01223 567808 Fax: 01223 567801

Email: support@ant.co.uk



# Contents

## Introduction 5

- Licence agreement 5
- About this User Guide 7
- Packing list 7

## Before you start 9

- What is OmniClient? 9
- Which networks are supported? 9
- Terminology 10

## Getting started with OmniClient 13

- Installing OmniClient 13
- Enabling automatic start up 14

## Using OmniClient 17

- Displaying available servers 17
- Mounting a file server 18
- Network printing 21
- Icon bar actions 25
- Filer functions 26
- Quitting OmniClient 27

## Configuring OmniClient 29

- Desktop boot file and OmniClient 29
- Startup file 31
- Mounts file 33

## Protocols 39

- Acorn Access/Access+ 39
- LanManFS 40
- NFS 41
- AUN Level 4 42

## !BootNet 45

## !Internet 47

- TCP/IP concepts 48
- Existing TCP/IP networks 48

Standalone NFS networks 51

## Installing !Internet 53

Configuration files 53

Different ways to configure the software 54

Installing TCP/IP within OmniClient 56

Advanced installation 61

## Using !Internet 63

Setting an IP address 63

Running !Internet 64

Internet module \* Commands 64

Absolute programs 65

Ethernet driver module \* Commands 66

RouteD \* Commands 66

## File mapping 91

Using extensions 91

Extensions file format 92

Flags 94

NFS file mapping 95

File mapping from RISC OS to UNIX 95

File mapping from UNIX to RISC OS 99

Editing the extensions file 103

## NFS star commands 105

## LanMan star commands 113

## Index 123



# 1 Introduction

THIS User Guide is intended to help you set up and use ANT OmniClient® on your network.

## Licence agreement

When you install or use this copy of OmniClient you are agreeing to our licensing terms as stated below. PLEASE READ THIS CAREFULLY BEFORE USING ANT OmniClient®.

### Copyright notice

Neither the whole nor any part of the OmniClient™ software nor its documentation may be reproduced, transmitted, transcribed, stored in a retrieval system or translated into any language in any form or by any means electronic, mechanical, optical, chemical, manual or otherwise without the prior express permission in writing of ANT Limited.

© Copyright ANT Limited 1995. All rights reserved.

### Terms of licence

ANT Limited provides OmniClient® under the auspices of worldwide copyright law and licenses its use to the purchaser. This means that unless you have ANT Limited's permission in writing to the contrary you may:

- Make one copy of OmniClient® for backup purposes so long as that copy is not operated at the same time as the original.
- Use the OmniClient® Single User Licence on a single computer, or move the program and use it on another computer but you MAY NOT use the program on more than one computer at once.



- Use the OmniClient® Site Licence on as many computers as are present at the postal address for which the Site Licence is registered so long as all the locations at which the computers are used share the same Post Code.

You MAY NOT make any copy of the documentation. Separate copies of the documentation are available to Site Licence users for a fee upon request to ANT Limited.

Violation of these terms of licence is likely to make you liable to criminal prosecution under worldwide copyright laws. If you are in any doubt as to your rights in respect of this licence you should contact ANT Limited for clarification.

If you prefer not to be bound by the terms of this licence you should return the software and documentation to ANT Limited for a refund before using it.

## **Disclaimer and Warranty**

ANT OmniClient® is sold on an “as is” basis and no warranty is made, intended nor implied with respect to the software or its documentation as to its completeness, merchantability, reliability or fitness for any purpose.

ANT Limited will not be held liable for any loss or damage resulting from the use or inability to use the software or its documentation.

Your statutory rights are not affected.



# About this User Guide

This user guide is intended to help all users of OmniClient. However, some of the program's features will only be of interest to system managers and administrators. Therefore the guide has been separated into two sections, so that the more complex technical information can be presented separately.

The first section is a quick start section, showing you:

- How to install OmniClient.
- How to begin using it.

The second section, starting on page 39, contains more detail on:

- Configuring and customising OmniClient.
- Exploiting OmniClient's powerful features.
- Details of each protocol supported.
- Reference guide for tools supporting each protocol, including !Internet, !BootNet, NFS and LanManFS and their star commands.

If you are a network administrator using OmniClient you should read this second section, particularly those parts of it which refer to servers which you have on your network.

## Packing list

Your copy of ANT OmniClient contains the following items:

- Software disk.
- User guide.
- Registration form.

If any of these items appear to be missing or damaged, please contact your supplier.

# About this User Guide

This user guide is intended to help all users of OmniClient. However, some of the program's features will only be of interest to system managers and administrators. Therefore the guide has been organized into two sections, so that the more complex technical information can be presented separately.

The first section is a quick start section showing you:

- How to install OmniClient

- How to begin using it

The second section, starting on page 59, contains more detail on:

- Configuring and customizing OmniClient

- Expanding OmniClient's powerful features

- Details of each protocol supported

- Additional guide for users supporting each protocol

- Additional information (Appendixes A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, and Index)

that are necessary.

If you are a network administrator using OmniClient you should read this second section, particularly those parts of it which refer to servers which you have on your network.

## Package list

Your copy of ANI OmniClient contains the following items:

- Software disk

- User guide

- Registration form

If any of these items appear to be missing or damaged, please contact your supplier.



## 2 Before you start

THIS chapter explains what OmniClient is, what you need to use it, and also some of the technical terminology which is used to describe what OmniClient does.

### What is OmniClient?

ANT OmniClient is a universal desktop filer for Acorn RISC OS computers, which allows Acorn platform users to store and retrieve files both from Acorn servers and those running different software on different platforms. All servers supported are accessed using the same interface, making it easy for the user to get the most out of a wide range of filing systems.

OmniClient creates a view on your desktop of your network as if it were a giant hard disc, with icons representing servers taking the place of icons representing folders or directories. You can even set the program up so that servers, or specified directories on them, are loaded on to your icon bar at start up.

### Which networks are supported?

OmniClient supports the following filing systems, although not all filing system modules may be available to you. Remember that any servers will need to be correctly set up with the appropriate software before you can connect. For example, you should ensure that Windows computers are running an appropriate version of Windows and not a stand-alone version.

At least one of the core OmniClient modules is supplied with all copies of the program. Additional modules may be purchased separately.

## Core modules

- Acorn AUN Level 4
- Acorn Access and Acorn Access+
- NFS (TCP/IP Protocol Suite) for Unix and Netware
- Lan Manager (NT workstation 3.1 and 3.5, NT Server 3.5, NT Advanced Server 3.1 and Windows for Workgroups), IBM OS/2 and OS/2 Warp

## Additional modules

- AppleTalk
- Research Machines NetLM
- Novell Netware

## Terminology

This user guide aims to keep technical jargon to a minimum, but there are some terms which cannot easily be avoided in order to explain the use of OmniClient clearly.

|                      |  |
|----------------------|--|
| <b>Alias</b>         | The short name by which a mount is referred to by OmniClient. The alias is the name which appears in filer windows and menus and beneath a mount icon on the icon bar.   |
| <b>Client</b>        | When you establish a mount on a server your machine is its client.   |
| <b>Filer</b>         | The windows in which the applications, directories and files available on a particular filing system are displayed on the RISC OS desktop.   |
| <b>Filing system</b> | Software which allows a particular kind of server to be mounted.   |
| <b>Mount</b>         | Used as both a verb and a noun. To mount is to establish contact with a server so that it (or the specified part of it) appear on the desktop or icon bar as standard filer items. A mount is the result of this process. When you have finished using the mount and close the filer window, you dismount. |
| <b>Mount point</b>   | Some large servers may have several locations in the file  |



directory structure where you can access them, to help you reach a set of files more directly or to restrict access to certain parts of the server. These access points are known as mount points.

**Protocol**

The specific computer language used by each kind of networking software to facilitate communication between computers. OmniClient supports multiple protocols allowing you to communicate with different types of networking software.

**Server**

Any machine on the network which you access using OmniClient. Most servers contain files and applications, but some are dedicated to a specific purpose, for example storing files which are waiting to be printed, in which case they are known as a print server.

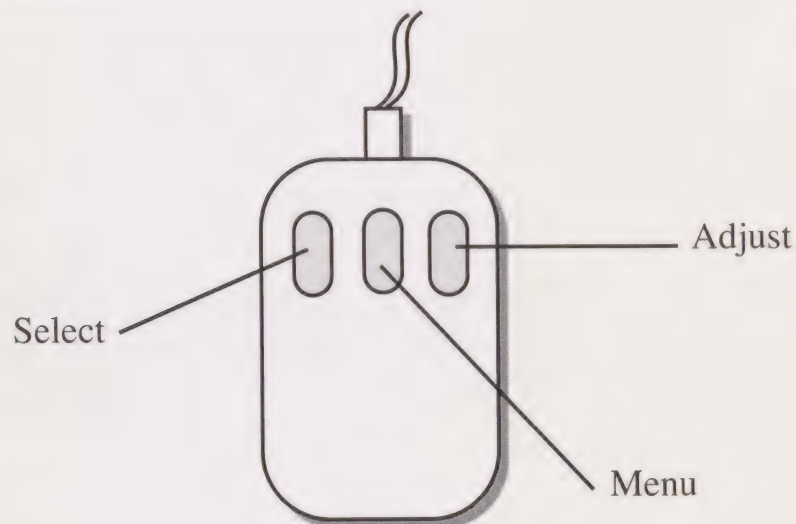
**Share**

Another name for a mount, used by Microsoft operating systems such as Windows for Workgroups and Windows NT.

Please also note the following:

**Mouse buttons**

Acorn computers have three-button mice, and the buttons each serve different functions. In many cases, which button you click decides what action takes place, especially clicking on icon bar icons where the Adjust (right) button is often used to provide alternative or additional functions.







# 3 Getting started with OmniClient

THIS chapter describes how to install OmniClient on your computer and configure it for use on your network. It should be read by network managers who are installing OmniClient and users of the single-user version of OmniClient who need to configure the program themselves.

Users who are simply using an existing OmniClient installation need not read this chapter.

## Installing OmniClient

This section describes how to install OmniClient on your computer so that it is ready to use.

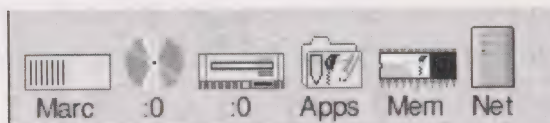
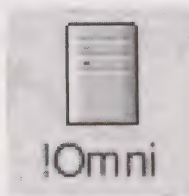
To load OmniClient on to your computer:

- 1 Copy OmniClient to your hard disk, or the network hard disk where your boot files are kept.

You can put the program where you like, for example in your Applications directory.

- 2 Find the !Omni program and double-click to load it on to your icon bar.

OmniClient is now ready to use. You will see the Net icon appear on the left edge of the icon bar. Other network icons may disappear:



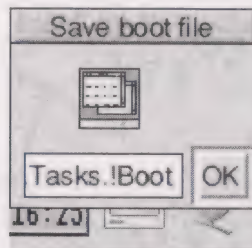
- 3 Click Select once on the Net icon.

OmniClient will scan your network and display a filer window listing all the servers which it finds.

## Enabling automatic start up

The above simple procedure is all you need to do in most cases. However, you may wish to add OmniClient to your desktop boot file so that it is loaded on to your icon bar whenever you start up your computer.

- 1 Reset your computer.
- 2 Find the !Omni program and double-click to load it on to your icon bar.
- 3 Save a desktop boot file by choosing **Desktop boot** from the Task menu:



- 4 Click **OK** in the Save boot file dialogue box. This will automatically save a new boot file in the correct place.
- 5 OmniClient will now load automatically whenever you switch on your computer.

If you take advantage of this facility, you will probably want to configure OmniClient carefully, so that it not only automatically loads the OmniClient but logs you in to specified servers as well.

## Customising OmniClient

Once you have installed OmniClient, there are many ways in which you can customise it to make it simpler to use, to reduce memory consumption and to automate the process of mounting servers.

- Save a list of mounts you use often so you do not have to



enter details each time you log on to a server.

- Specify a list of mounts to log on to automatically at startup.

Network managers can also use OmniClient's various options to restrict users' access to file servers to particular mount points, or even to hide file servers from users.

- Configure the OmniClient Startup file so that only the network filing systems available on your machine are enabled (see *Configuring OmniClient* on page 29 below). This will help to save memory.





# 4 Using OmniClient

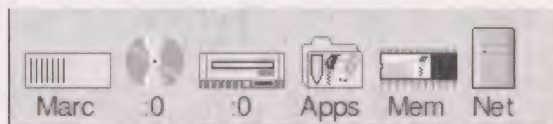
THIS chapter describes how to use OmniClient to display and mount file servers.

Once you have installed OmniClient you are ready to use it to log on to servers on any networks to which your computer is attached and for which you have the relevant OmniClient protocol modules.

## Displaying available servers

In most cases, your copy of OmniClient will be set up so that it loads automatically when you boot up your computer. If not, you will need to locate the !Omni application and double click on it.

Whichever route you take, the OmniClient Net icon will be displayed at the left side of the icon bar along with other discs and filing systems:



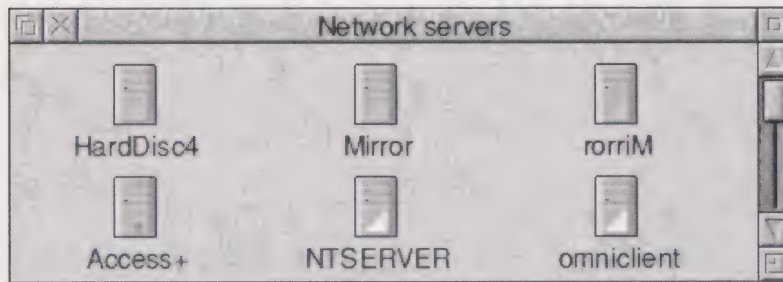
- 1 Click Select on the Net icon.

OmniClient will search your network for available servers, and display a window containing their names and types.

### The network servers window

The network servers window lists all the servers which OmniClient can find on networks to which your computer is

attached. Each server is shown by an icon:



- In Small icon and Large icon views, server types are distinguished by icons with differing corner flashes.

|               |                              |
|---------------|------------------------------|
| NFS           | Dark blue flash              |
| Windows       | Light blue flash             |
| Level 4       | Yellow                       |
| Acorn         | Green flash                  |
| Acorn Access+ | Green flash with a plus sign |

- In **Full Info** view, the servers' full names and type are shown along with other information such as IP address, host name or Ethernet ID.
- You may sort the list of servers by name or by protocol. Choose **Display** from the **Filer** menu and choose **Sort by name** or **Sort by Protocol** as required.

## Mounting a file server

There are several ways to mount a file server using OmniClient.

- Your copy of OmniClient may be configured to mount a server automatically as part of its boot sequence. See *Desktop boot file and OmniClient* on page 29 for further details.
- By choosing a preconfigured mount from the icon bar Mounts menu. See *Mounts file* on page 33 for further details.
- By entering details of a new mount via the icon bar mounts menu.



- By clicking on a file server icon in the network servers window.

## Mounting a file server from the File servers window

To mount a file server from the File servers window.

- 1 Double-click on the required file server icon.
- 2 A log on window for the type of file server will appear, unless multiple mount points are available for the server – see 3 below.

The details required differ for each kind of file server, and depending on your setup some details may be filled in already.

| Mount 'NFS' server   |                   |
|--|-------------------|
| Name   | ant/zebedee       |
| Server name  | zebedee.ant.co.uk |
| Directory path   | /extra/ant        |
| User name  | ant               |
| Password   |                   |
| Authenticator  | zebedee           |
| <input type="button" value="Cancel"/> <input type="button" value="Connect"/> |                   |

NFS servers may need the name of the name server acting as network authenticator, if it is not the main server.

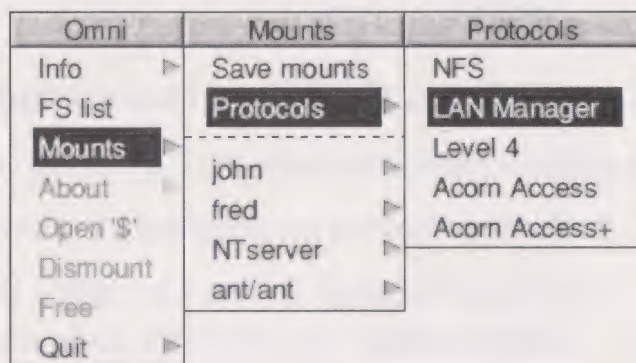
- 3 On some file servers several mount points may have been set up, making it possible to log into different directories. In these cases, choose the directory folder you want from the window which appears, and double-click on it. You will then see a log on window.

## Mounting a file server from the icon bar menu

If the mount you want is not listed by name:

- 1 Choose **Mounts** from the icon bar menu.

## 2 Display the **Protocols** submenu:



## 3 Choose the protocol of the mount you want.

A blank log-in window will be displayed for you to fill in and log on to the server.

## 4 Click **Connect** to log on to the server.

To mount a file server from the icon bar menu:

### 1 Choose **Mounts** from the icon bar menu.

A menu will appear listing all the mounts in your Mounts file (see *Mounts file* on page 33).

### 2 Display the Mount server dialogue for the mount you require by sliding off its entry or double-clicking on it.

### 3 Complete any details still required on the Mount server dialogue and click **Connect**.

If all the required details were already stored, you will not see a dialogue but will be logged on when you double-click the server's name.

## Mounting a server automatically

If your copy of OmniClient has been set up to mount servers automatically, they will be located and a connection made when you start up your computer.

- If the mount needs no password, or the password has been stored by OmniClient, OmniClient will access the mount immediately.
- If a password is needed OmniClient will display a Mount dialogue box for you to enter the password.



## Using mount aliases

The mount alias is the name which appears under the OmniClient icon on your icon bar when you are logged on to a file server. It is defined in the Name field of the Mount server dialogue box.

In many cases a default alias will be shown, usually the short form of the server name. If a mount of this name already exists, you will need to enter a different name.

Each mount must have a unique name. If you try to use a name which is already in use, OmniClient will prompt you to enter a different name.

## Mounting additional servers

Once you have mounted a fileserver, the icon bar icon changes to represent the mounted file server. Clicking on it will display the directory it mounts rather than the server list.

To display the list of file servers or mount additional file servers, you must choose FS List from the icon bar menu of the mounted server.

## Network printing

In addition to accessing files stored on servers across networks, OmniClient lets you use network printers.

- You should be aware that there are many possibilities for problems when printing across networks. Some potential sources of difficulty are font problems, PostScript/page description language incompatibilities, failure of communication between printer and server, network hardware problems.

There are two ways in which you can print across networks.

- You can send the document as a file, using the file option in !Printer. You should do this for non-Acorn networks other than NFS. This option will eventually be superseded by an updated version of !Printers which will include improved support for OmniClient printing.

- If you are using NFS, you can use the NFS option in !Printers.

## File printing

To print a document as a file:

- 1 Load !Printers:
- 2 Click menu on the printer icon and choose **Printer control**. If necessary, add a new printer to the list, for example a PostScript laser printer.
- 3 Click menu on the printer name and choose **Connections**. A dialogue box will appear:

Connections

LW II NTX

Print in background ☐

☐ Parallel

☐ Serial

Baud rate: 1200

Data bits: 8 bits

Parity: None

Stop bits: 2 bits

X-On/X-Off ☐

☐ Net

☐ NFS

Server:

Printer:

Username:

Options:

☒ File

ap.ScrapDirs.ScrapDir.Printout

Append to file ☐

☐ Direct drive

Cancel OK

- 4 Click on the **To file** radio button.
- 5 Enter a filename of the following form:

`OmniPrint#FSName;Server;Printer;User;Password;Options;Size:`

- 6 Save the new !Printers configuration.



## Configuration details

You must enter:

**FSName**            Name of the filing system  
**Server**            Name of the server  
**Printer**           This may be the same as the server name, for  
                         example with a stand-alone AUN printer)

**User, Password, Options** and **Size** are all optional fields.

The exact combination of fields is protocol-specific.

### Examples

```
OmniPrint#NFS;ant;lp;nas:
```

```
OmniPrint#Net;MATRIX;MATRIX:
```

```
OmniPrint#LanMan;NTserver;NTprint;guest;guest:
```

For protocols that require the file size prior to the transmission of any data, an optional field **size** can be included in the special field.

Any filename given is ignored – the critical data is contained in the special field. Fields are semi-colon separated, and you should enter any illegal (RISC OS) characters as follows:

|              |                  |
|--------------|------------------|
| <b>SPACE</b> | ~_ (underscore)  |
| "            | ~'               |
| (solidus)    | ~1 (lowercase L) |
| :            | ~1 (numeric one) |
| ;            | ~!               |
| ,            | ~.               |
| ~            | ~~               |

## Printing using NFS

You can use the NFS software with the standard RISC OS Printers application to print on printers connected to remote NFS servers.

- 1 Set up OmniClient to use NFS before you use the Printers application. This will make sure that it uses NFS software

as required.

- 2 Set up the printer as usual, but make the following entries in the Connections window:

Connections

PostScript

Print in background ☐

☐ Parallel

☐ Serial

Baud rate 1200

Data bits 8 bits

Parity None

Stop bits 2 bits

X-On/X-Off ☐

☐ Net

☒ NFS

Server ant

Printer lp

Username nas

Options

☐ File

ap.ScrapDir.ScrapDir.Printout

Append to file ☐

☐ Direct drive

Cancel OK

- 3 Choose **NFS**.
- 4 Enter the name of the **server** (this must be running the **pcnfsd** daemon, also used to enable the server to act as a name server).
- 5 Enter the name of a **printer** that the server can access. If you don't enter a name, the server will use printer **lp**, the default name for a UNIX printer.
- 6 Enter the **username**. The program doing the printing (**lpr** in UNIX) uses this to establish your access rights. If there are no restrictions on printer use on your network, you need not enter your user name.
- 7 Enter any options you know to be required. Check with your system administrator for any details which may be needed.



- 8 Choose **Save choices** from the Printers icon bar menu to save this setup.

Once you have set up your printer like this, you will be able to print to the remote network printer as if to a local RISC OS printer.

## Icon bar actions

Clicking on the icon bar Net or mount icon can have several results, depending on whether a server is mounted or not.

|                      |                |                       |
|----------------------|----------------|-----------------------|
| <b>No connection</b> | Select, Adjust | Show network servers  |
| <b>Connected</b>     | Select         | Show user files (URD) |
|                      | Adjust         | Show network servers  |

*Expert mode* Switching to expert mode enables a number of extra short cuts. The results will again depend on whether a connection was established or not.

|                      |                        |                                 |
|----------------------|------------------------|---------------------------------|
| <b>No connection</b> | Select                 | Show network servers            |
|                      | Adjust                 | Show empty mount dialogue boxes |
|                      | Shift-select           | Show network servers            |
| <b>Connected</b>     | Select                 | Show user files (URD)           |
|                      | Adjust                 | Show empty mount dialogue boxes |
|                      | Shift-select           | Show network servers            |
|                      | Ctrl-select drag       | Copy URD files to a filer       |
|                      | Shift-ctrl-select drag | Move URD files to a filer       |

In most protocols, URD, the user root directory, is the same as the mount's root directory. However AUN NetFS requires both actions to be available separately.

Most protocol modules do not have a separate User Root Directory (URD) /Root Directory (\$) concept, and for these an 'Open URD' and an 'Open Root' action work the same, opening

a filer view of the mount root.

- In expert mode you can also drag files from a filer window to the icon bar icon of a connected mount to move the files to the URD of the mount.

## Filer functions

These options are available from the icon bar menu.

### Getting information about a mount

Choose **About** from the icon bar window. A dialogue box will appear listing the mounts name, server name, directory path, user name and authenticator for the chosen mount.

The authenticator is the name server which may be used by NFS servers.

### Opening a root directory

Choose **Open '\$'** from the icon bar menu.

### Dismounting

Choosing **Dismount** dismounts the file server and closes any windows from files stored on it.

*Expert mode* If multiple connections are established, choosing Dismount from the icon bar menu produces a submenu.

- Click Select on **Dismount** on the main icon bar menu, or **This** on the Dismount submenu, to dismount the selected mount.
- Choose **All** from the Dismount submenu to dismount all current connections.

### Finding the free space on a mount

The option **Free** opens a dialogue box which shows you the amount of free space on a mount.



# Quitting OmniClient

To dismount a single server, choose **Dismount** from the icon bar menu of that server.

To quit OmniClient:

- Click on the Task Manager icon on the icon bar.
- Click Menu over the OmniClient entry.
- Choose **Task/Quit**.

*Expert mode* Expert mode permits you to quit OmniClient directly from the server icon bar menu. Two options are available from the Quit menu:

- Choosing **Filer** closes all Filer windows but retains the underlying network connection.
- Choosing **All** closes down OmniClient and the underlying network connections.





# 5 Configuring OmniClient

THIS section describes how to configure OmniClient.

To allow configuration to be stored on a per-user basis, all configuration files used by the OmniClient application are accessed through the system variable <Omni\$Path>, which is by default set to !Omni.Files.

This directory contains four files:

- Mounts (see *Mounts file* on page 33)
- Extensions (see *File mapping* on page 91)
- Startup (an obey file)
- StartShare (an obey file for Acorn Access)

OmniClient information is also stored in the desktop boot file.

## Desktop boot file and OmniClient

It is possible to use desktop boot files to simplify starting up OmniClient and logging on to the servers which are most commonly used.

### Specifying mounts for loading at startup

When the OmniClient application is started, the command to start the application which is stored in the desktop boot file may be followed by a list of mount alias names, for example:

```
Run adfs::HardDisc4.$apps.!Omni alex@ant mymount@server example
```

Each name is looked for in the mounts file, with the following consequences:

- If the mount is with a filing system (or server) that needs no password, or if the password is in the mounts file, the mount will be accessed immediately and an icon will be

placed on the icon bar.

- If the mount requires a password a static login dialogue box is brought up for the mount and you should type in the password to complete the login process.

Instead of a list of mount aliases the OmniClient command may simply be followed by the string **-ALL**, in which case OmniClient will load all the mounts in the mounts file:

```
Run adfs::HardDisc4.$apps.!Omni -ALL
```

To save a list of mounts to connect to at startup, see *Saving a mounts file* on page 33.

You can change your OmniClient setup at any time by saving the Desktop Boot file again.

You can also edit the command by hand. To edit this command:

- 1 Load the desktop boot file into a editor such as !Edit. It is a good idea to make a backup copy of the original file first, in case you make any mistakes or change your mind.
- 2 Type in the changes you want.
- 3 Save the file.
- 4 Restart your computer and reload the program for the changes to take effect.

The command **\*OmniMount** provides an alternative means of starting up mounts from a command file. The command can either list individual mounts by their alias names, or load the entire mounts file with **-ALL**.

```
*OmniMount alias1 alias2
```

The OmniMount command should be placed at any point in your desktop boot sequence after !Omni has been loaded.

## Supporting multiple mounts files

To support multiple saved mounts files (for example a master network file for booting, and a writable per-user mounts file), a \*command **OmniLoadMounts** lets a named mounts file be loaded (merged with those already in memory). Mounts will always be saved to the mounts file <Omni\$Path>Mounts unless



a mount has the LOCKED flag set (see below).

The OmniClient application responds to the 'desktop save' message by storing a command that will restart OmniClient with the current setup if this is possible. To this end the application may raise a dialogue box asking the user if the unsaved mounts and their aliases should be written to the mounts file.

## Security warning

If network security is of concern to you, do not automate log ons by including your password or network identity in your boot file. You may prefer to be confident of your security and type in IDs and passwords as required. When you save a mounts file, passwords are not included. You can only enter passwords manually.

## Startup file

The Startup obey file is executed when OmniClient is loaded, and may contain commands to load protocol modules, set Omni\$User variables, and turn various functions on and off.

## Configuration options

The startup file can set the system variable Omni\$Options which allows the default configuration of the application in a similar manner to that of Acorn !Edit and Edit\$Options. Options are:

| Option    | Action   |     | Result        |
|-----------|--|-----|---------------|
| <b>sN</b> | Sort type  | N=1 | Name          |
|           |  | N=2 | Protocol/Name |
| <b>dN</b> | Display type   | N=0 | (Large icons) |
|           |  | N=1 | (Small icons) |
|           |  | N=2 | (Full info)   |
| <b>x</b>  | Expert mode on – some extra menu items and iconbar-click functionality |     |               |
| <b>a</b>  | Auto-location of network servers on                                    |     |               |

For example:

```
Set Omni$Options s1 d1 a
```

- The **a** flag automatically locates servers, so all servers accessible on the network will be listed in the Mounts menu.
- If you are using LanManFS and are connected to, for example, a Windows for Workgroups network, automatic location can mean that OmniClient will list every machine on the network as a server. Auto-location of LanMan servers must be turned on using the \*LMLogon command. See page 118 for further details.

*Expert mode*

- The expert mode **x** flag is only available with ANT OmniClient, and not with versions of OmniClient shipped by Acorn. For details of expert mode options see sections throughout this user guide flagged with an *Expert mode* sidehead.

## Loading protocol modules

You may use the startup file to specify which protocol modules to load. Not loading modules you don't need will save memory.

To specify modules to load:

- 1 Load the startup file into an editor.
- 2 Go to the section in the file:

```
SetEval Omni$LanMan      1
SetEval Omni$Net         1
SetEval Omni$Share       1
SetEval Omni$NFS         1
```

- 3 For those protocol modules which you do not wish to load, replace the 1 with 0.
- 4 Save the file and restart the computer. The next time OmniClient loads, it will not load the modules you turned off.

## Setting the user name

The system variable Omni\$User stores a user name to use in



mount dialogue boxes. You can store one name, or specify the filing system for which a particular user name should be used.

For example:

**Omni\$User TJones**

enters the user name TJones in any mount dialogue boxes, while

**Omni\$UserNFS TJones**

enters the user name TJones only in NFS mount dialogue boxes.

Options are:

- NFS
- LanMan
- Net
- Share

## Mounts file

This section describes how to store details of servers on your network by setting up a mounts file.

Each line of the mounts file contains information for a mount. Usually this will be all the information needed to connect a mount with the exception of the user's password, although this can be supplied if the user feels it is not a security problem.

### Saving a mounts file

The easiest way to create a mounts file, and the way which you should initially use, is to do the following:

- 1 Log on to the mounts you wish to enter in the mounts file.
- 2 Display the **Mounts** menu from the icon bar menu.
- 3 Choose **Save mounts**.
- 4 The current mounts will be added to the mounts file and next time you start !Omni they will appear on the mounts menu.

Note that saving mounts like this does not save password information.

You may add mounts to the mounts file at any time by repeating this process. Choosing Save mounts adds mounts to the existing list rather than over-writing existing entries.

## Manually editing the mounts file

It is possible to edit the mounts file by hand, but it is usually unnecessary to do so. If you want to save the details of a new mount to add it to the file simply choose Save mounts again when the mount is connected.

To create an entry in the mounts file:

- 1 Load the OmniClient mounts file (in **!Omni.Files.Mount**), into an editor such as !Edit.
- 2 Enter the details for the mount using the following syntax:

`Protocol, Alias, Flags, Server, Mount path, User ID, Password, Authentication`

**Protocol** is the network protocol.

**Alias** is the mount alias as described on page 10.

**Flags** are described below. They optionally hide, lock or force the display of server details.

**Server** is the actual name of the server.

**Mount path** is the mount path for the mount point on the server.

**User ID** is the user name.

**Password** is the password. To leave this field blank, do not type anything between the commas. The user will have to type in their password to complete the mount process when the mounts file is used.

**Authentication** is the name of the network authenticator/ name server and is only required for NFS mounts.

Separate entries with a comma.

- 3 You may enter comments in the mounts file by prefixing



lines with a # character, for example:

#### **# Mount list**

Comments are overwritten when the mounts file is resaved. A comment line containing a datestamp is automatically generated.

- 4 On startup, OmniClient will check the mounts file for the mount name.

Below is an example mounts file:

```
# > <Omni$Path>Mounts
# Mount list saved by OmniClient at 16:14:54 on 15 Nov 1994
#
# Protocol, Alias, Flags, Server, Mount path, User ID, Password, Authentication
#
Access,Thomas,,,,Thomas
LanMan,IanC,L,PC1,C:,ian
LanMan,PC2,L,PC2,,ian
Novell,Scot,HL,PC3,D:\exported,fred
NFS,Apps,P,zebedee,/extra/apps,apps,letmein
NFS,News,P,zebedee,/scsi2/news,borris,,pollux
```

In the above example the mounts called Thomas and Apps could be mounted without typing a password (the former does not need one, the latter has the one it needs). Furthermore, null passwords are supported at two levels:


- Putting two commas (as above in the mount News) forces the user to enter a password.
- Putting empty double quotes within the commas causes an empty password to be used to login without prompting the user.

## **Removing mounts from the mounts file**

To remove a mount from the mounts file:

- 1 Choose it from the Mounts menu.

2 Display its log on window:



| Mount 'LAN Manager' server       |          |
|----------------------------------|----------|
| Name                             | NTserver |
| Server name                      | NTSERVER |
| Directory path                   | public   |
| User name                        | guest    |
| Password                         | ----     |
| Authenticator                    |          |
| <div>Delete Cancel Connect</div> |          |

3 Click on the **Delete** button. The mount will be removed from the list.

4 Choose **Save mounts** from the Mounts menu to prevent the deleted mount from re-appearing.

## Flags in the mount file

Three further flags let you store additional information about specific mounts, altering the way OmniClient handles them.

These flags are for use by network administrators who need to set up systems where users get access to servers which can not be located automatically, or to hide servers from users.

The three flags are:

- L Locked
- P Preset
- H Hidden

To set these flags edit the mounts file. Note that Preset and Hidden are mutually exclusive.

- Mounts with the Locked flag do not get saved into the user mounts file, and so can be used in read-only, master mounts files used at boot time.
- Mounts with the Hidden flag set do not appear in the mounts list submenu, in the 'servers' window or on the iconbar. The Hidden flag is provided to allow for 'system' mounts which are used in a networked boot sequence (for example a scrap area) but which remain hidden from the



user and do not appear on the icon bar. In conjunction with the Locked flag for master mounts file(s), a network system administrator can setup a very flexible boot sequence that has its complexity hidden from the end user.

Note that the Hidden flag will not prevent servers/mounts appearing in the 'servers' window that have been auto-located by a network scan (if this is enabled – see *Startup file* on page 31).

- The Preset flag allows extra servers/mounts to be presented in the network servers window – perhaps non-local servers which protocols cannot automatically scan for. Mount file entries with the Preset flag set do not appear in the mounts list submenu unless they are currently connected. This flag has a higher priority than Hidden, and mounts file entries that mistakenly have both set, will be treated as Preset.





# 6 Protocols

THIS chapter lists the network protocols supported by OmniClient and describes how they work with the program, the special options they may require and any particular difficulties you may have.

Each section also explains which of the following technical reference sections may be useful in setting up OmniClient to work with the protocol on your network.

## Acorn Access/Access+

This is the protocol which handles native Acorn servers.

Details of these servers and the utilities which support their use can be found in the Acorn documentation which came with them.

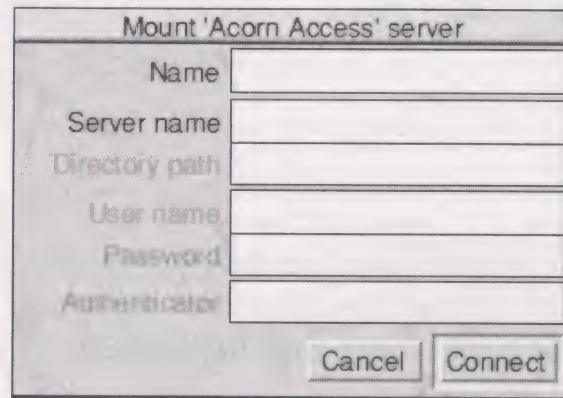
To access these servers you must have the following software in place:

- !Internet
- Acorn Access or Access+

### Logging on to Access/Access+ servers

When you log on to an Access/Access+ server, you will see the

following dialogue box:



A screenshot of a graphical user interface dialog box titled "Mount 'Acorn Access' server". The dialog box contains six text input fields arranged vertically, each with a label to its left: "Name", "Server name", "Directory path", "User name", "Password", and "Authenticator". At the bottom right of the dialog box, there are two buttons: "Cancel" and "Connect".

You will need to enter the following:

- Name (alias)
- Server name
- Password (Access+ only)

## LanManFS

This is the protocol module which handles Microsoft Windows and networking connectivity. It allows you to connect to servers under the following programs:

- Windows NT
- Windows for Workgroups
- LAN Manager
- IBM OS/2 and OS/2 Warp

To help manage these programs, OmniClient uses a filing system called LanManFS, which has its own star commands. These are listed in the chapter *LanMan star commands* on page 113.

Because every computer on a Windows for Workgroups network is potentially a share (or mount), you can configure LanManFS so that it does not auto-locate servers and thus present a quite unwieldy Network servers window. See page 118 for details.

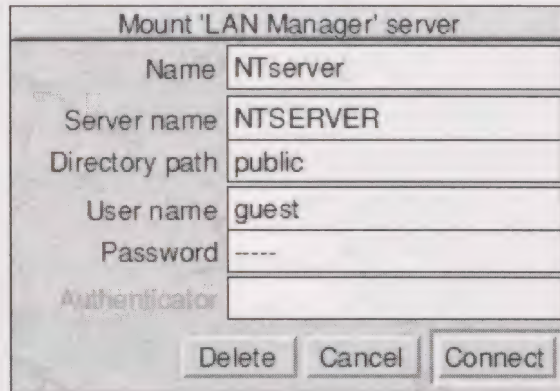
File names used with LanManFS will also require translation from Acorn-specific forms to a format which is compatible with PCs. See the section *File mapping* on page 91 for further details.



To access these servers you must have a network driver module loaded (for example, Ether3 for Ethernet). You can do this most simply by installing !Internet. If you only want to use LanManFS and no other OmniClient protocols, you can save memory by loading just the driver module without the rest of !Internet.

## Logging on to Lan Manager servers

When you log on to a Lan Manager/Windows server, you will see the following dialogue box:



A screenshot of a Windows-style dialog box titled "Mount 'LAN Manager' server". It contains several input fields with labels to their left: "Name" (containing "NTserver"), "Server name" (containing "NTSERVER"), "Directory path" (containing "public"), "User name" (containing "guest"), "Password" (containing "----"), and "Authenticator" (empty). At the bottom right are three buttons: "Delete", "Cancel", and "Connect".

| Mount 'LAN Manager' server   |          |
|--|----------|
| Name   | NTserver |
| Server name  | NTSERVER |
| Directory path   | public   |
| User name  | guest    |
| Password   | ----     |
| Authenticator  |          |
| <input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Connect"/> |          |

You will need to enter:

- Name (alias)
- Server name
- Directory path (share name)
- User name
- Password

## NFS

This is the protocol module which supports UNIX networks.

To help manage this, OmniClient uses a series of programs including !Internet, NFS and OmniNFS. You will need to set up !Internet and other software components if you need to access NFS file servers. To do this, see the following chapters:

- *!Internet* on page 47
- *Installing !Internet* on page 53

- *Using !Internet* on page 63
- *NFS star commands* on page 105

There is also a file name mapping issue discussed in the section *NFS file mapping* on page 95.

## Logging on to NFS servers

When you log on to an NFS server you will see the following dialogue box:

| Mount 'NFS' server   |                   |
|--|-------------------|
| Name   | alex/zebedee      |
| Server name  | zebedee.ant.co.uk |
| Directory path   | /home/alex        |
| User name  | alex              |
| Password   |                   |
| Authenticator  |                   |
| <input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Connect"/> |                   |

You will need to enter the following details:

- Name (alias)
- Server name
- Directory path
- User name
- Password
- Authenticator (server name, where applicable)

## AUN Level 4

This is the protocol used by Acorn AUN networks.

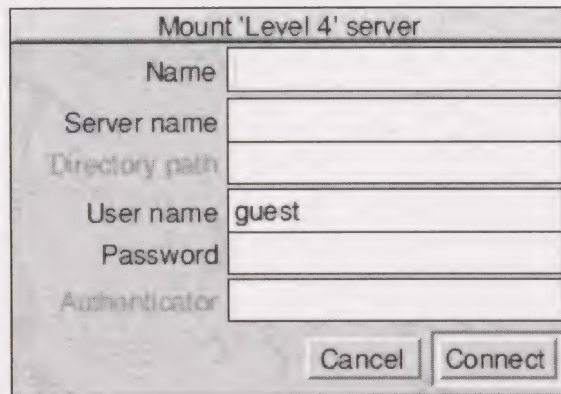
!BootNet must have been loaded before using this protocol. See *!BootNet* on page 45 for further details.

## Logging on to AUN servers

When you log on to an AUN server you will see the following



dialogue box:



A screenshot of a Windows-style dialog box titled "Mount 'Level 4' server". The dialog box contains several input fields: "Name", "Server name", "Directory path", "User name" (which contains the text "guest"), "Password", and "Authenticator". At the bottom right of the dialog box are two buttons: "Cancel" and "Connect".

You will need to enter the following details:

- Name (alias)
- Server name
- User name
- Password





# 7 !BootNet

!BootNet is an application that loads the software modules required to allow Level4 to be used over Ethernet.

!BootNet should be run either

- On start-up, to allow Ethernet access to Level 4 file servers.
- After !Internet is run, when Level 4 file servers are used on a configured TCP/IP network.

Normally this software is loaded automatically from the Ethernet card ROM or from system ROM, except when it is run after !Internet where the !BootNet application is required.

Where the software is resident on Ethernet card ROM, then the modules are loaded if the configuration requires it:

**\*Configure BootNet On** loads the modules from the Ethernet Card ROM, but on an Acorn Access card some of the modules are loaded even if BootNet is configured off.

- If you wish to access Level 4 and your Ethernet cards contain DCI2 modules, configure BootNet off and use the copy of !BootNet provided.
- If you are running a full TCP/IP network, communicating with NFS servers, then you should ensure that the file **!BootNet.!Configure** does not re-load the driver. See the comments inside that file for full information. The Release Note contains hints and tips about this.

# Index

1. Introduction  
2. The BootNet Project  
3. The BootNet Architecture  
4. The BootNet Software  
5. The BootNet Hardware  
6. The BootNet Network  
7. The BootNet Security  
8. The BootNet Performance  
9. The BootNet Future  
10. The BootNet Conclusion



# 8 !Internet

OmniClient enables you to connect Acorn RISC OS computers to a TCP/IP network, and to access computers on that network in a variety of ways. !Internet is the program which makes TCP/IP protocols available to RISC OS computers. NFS, the filing system which uses TCP/IP protocols, requires !Internet to be present in order to operate.

You can use !Internet and OmniClient NFS functionality to connect RISC OS computers directly to an existing Ethernet-based TCP/IP network. To do so, your RISC OS computers will need to have an Acorn Ethernet interface fitted.

## Finding out more

For general information on the use of a RISC OS computer and its desktop interface, see the *Welcome Guide* and *RISC OS 3 User Guide* supplied with it.

For details of how to use the TCP/IP Protocol Suite (Release 2), see the *TCP/IP Protocol Suite (Release 2) User Guide*.

For details of how to use the programming interfaces provided by the TCP/IP Protocol Suite (Release 2), see the *TCP/IP Protocol Suite (Release 2) Programmer's Guide*, available separately from Acorn Development Support. This includes a disc of useful C libraries.

You should also see any relevant documentation supplied with other computers on your TCP/IP network.

Finally, you can get more detailed information from *Internetworking with TCP/IP*. Douglas Comer (1988) Prentice-Hall, Englewood Cliffs, NJ, USA.

# TCP/IP concepts

When you install OmniClient, you will have to assign certain names and numbers to the computers on your TCP/IP network, and to their network interfaces. This section explains those names and numbers.

## Existing TCP/IP networks

If you've already got a TCP/IP network running on your site, you should already have naming and numbering schemes set up. Make sure that any names and numbers you assign conform to this scheme, and that you first contact anyone who administrates their allocation.

### Host names

Each computer on your network must have a *principal host name*, or *host name* for short. Your users will use this name to refer to the computer. The name must be unique on your site – you can't have two computers with the same name.

It helps your users if each host name is easy for them to remember. One way to do this is to use a theme, such as planets (e.g. **saturn**, **uranus**); another way is to give names that have some relationship to the computer's function on your network (eg **accounts1**, **accounts2**). You can combine these ideas – so you might name the graphics department's computers after famous artists (eg **turner**, **vangogh**).

### Interface names

Each network interface in each computer – whether it be an Ethernet or Econet interface – must also have an *interface name*. Again, this name must be unique on your site – you can't have two interfaces with the same name.

If there's only a single interface in a computer it's normal to use just the principal host name as the interface name. If there are two interfaces in a name it's normal to refer to the principal host name



in each interface name: so a machine named **saturn** may have interfaces named **saturn\_eco** and **saturn\_ether**.

## Internet addresses, netmasks and subnets

Furthermore, each interface must also have a unique numerical address, known as its *Internet address*. It is this address that the TCP/IP protocol uses to communicate; if a user specifies a host name or interface name, the software automatically converts it to an Internet address.

An Internet address is four bytes long. These four bytes are split into fields:

|                        |                                     |                     |
|------------------------|-------------------------------------|---------------------|
| High                   |                                     | Low                 |
| <i>network address</i> | <i>subnet address</i><br>(optional) | <i>host address</i> |

*Figure 8.1 Fields within an Internet address*

The *network address* identifies an entire network (which is typically a whole site). The *subnet address* is optional, and identifies a local network that forms part of the main network. The *host address* identifies a host on that network.

A *netmask* specifies the portion of the address used by the network and subnet addresses. For example, if the network address is held in the top byte, and no subnets are used, the netmask would be 0xFF000000 (i.e. FF000000 hexadecimal).

Unlike the interface name, the Internet address must be unique on all networks with which the interface will ever communicate.

## If you plan to connect to other sites...

If you plan to connect to other sites over the Internet, you need to ensure not only that Internet addresses are unique to your site, but also that they are unique to the entire Internet. The Internet already connects together thousands of sites, each with many hosts. Clearly it's impossible to keep so many Internet addresses unique on an informal basis.

Consequently there is an administrative body responsible for

allocating network addresses. Your Internet service provider will help you sort out these issues and will help you with the allocation of numbers and addresses.

If you are not using an Internet service provider, you should contact the following organisation before you use the Internet to connect to other sites; write or send email to:

*DDN Network Information Center  
SRI International  
Room EJ217  
333 Ravenswood Avenue  
Menlo Park, CA 94025  
USA*

email: HOSTMASTER@SRI-NIC.ARPA

Depending on the size of your network, you will be allocated a Class A, B or C address: these use respectively the top one, two or three bytes for the network address. It is your responsibility how you use the remaining unallocated bytes to specify subnets and hosts. For example, let's say you've been allocated a Class B network address, and so have two bytes free for your own use:

- If all your site's computers are connected to a single local network, you won't need to use subnets, and so might use all two bytes for the host address (allowing 64k hosts). In this case, you'd use a netmask of 0xFFFF0000.

Note that an Ethernet generally behaves as a single network, even if it is made up of multiple segments of cable (unless divided by gateways).

- However, if your site's computers are connected to different local networks (such an Ethernet and some Econets) you'll need to use subnets. You might decide to use 5 bits for the subnet address (allowing 32 subnets), and the remaining 11 bits for the host address (allowing 2000 hosts). In this case, you'd therefore use a netmask of 0xFFFFF8000.

Note that separate Econets (i.e. those not connected together by Econet bridges) form separate subnets.



# Standalone NFS networks

If you don't plan to connect to other sites over the Internet, all you need to ensure is that each interface's Internet address is unique on your own site. Acorn suggests you use the following scheme:

| <i>network address</i> | <i>host address<br/>(high byte)</i> | <i>host address<br/>(middle byte)</i> | <i>host address<br/>(low byte)</i> |
|------------------------|-------------------------------------|---------------------------------------|------------------------------------|
|------------------------|-------------------------------------|---------------------------------------|------------------------------------|

*Figure 8.2 Suggested local TCP/IP numbering scheme*

Number your local networks from one: for example, you might number your Ethernet as net 1, and an Econet as net 2. Likewise, number your hosts (not your interfaces) from one. Your available Internet addresses and their meanings would then be:

| <b>Ethernet</b> | <b>Meaning</b>       | <b>Econet</b> | <b>Meaning</b>            |
|-----------------|----------------------|---------------|---------------------------|
| 1.0.0.1         | host 1 on Ethernet   | 2.0.0.1       | host 1 on Econet          |
| 1.0.0.2         | host 2 on Ethernet   | 2.0.0.2       | host 2 on Econet          |
| 1.0.0.3         | host 3 on Ethernet   | 2.0.0.3       | <i>and so on up to...</i> |
| 1.0.0.255       | host 255 on Ethernet | 2.0.0.255     | host 255 on Econet        |
| 1.0.1.0         | host 256 on Ethernet | 2.0.1.0       | host 256 on Econet        |
| 1.0.1.1         | host 257 on Ethernet | 2.0.1.1       | <i>and so on...</i>       |

Of course, if a machine has only got one interface fitted, you'll only use one of the addresses assigned to it; one of the addresses will be 'wasted'. But if you later upgrade the machine to add a second interface, you'll already have a meaningful Internet address reserved for it.

## Physical addresses

Each interface also has a six byte *physical address* (alternatively known as its *MAC address*). You shouldn't need to do anything to set this up, because:

- An Ethernet interface's physical address is unique worldwide, and is set in the hardware at the time of manufacture.
- An Econet interface's physical address is based on its

network and station numbers. So long as you've correctly installed the Econet, these should be unique to your site.



# 9 Installing !Internet

IN THE earlier chapter *!Internet* on page 47 we outlined different ways in which you could set up OmniClient. You'll have to edit some configuration files to do so. This chapter tells you how to make those changes and install the software. It assumes you are using the desktop, and are familiar with simple use of it.

If you have any problems refer to the *RISC OS User Guide* supplied with your RISC OS computer.

## Configuration files

The configuration files you need to edit are held within the Internet application. Rather than refer to them all the time by their lengthy full pathnames, we'll just use the leafname. The files are supplied on the *Network* distribution disc as:

| Filename                                  | Leafname                |             |
|---|-------------------------|-------------|
| <code>\$.!Internet.!Configure</code>      | <code>!Configure</code> |             |
| <code>\$.!Internet.files.hosts</code>     | <code>hosts</code>      | } databases |
| <code>\$.!Internet.files.networks</code>  | <code>networks</code>   |             |
| <code>\$.!Internet.files.protocols</code> | <code>protocols</code>  |             |
| <code>\$.!Internet.files.services</code>  | <code>services</code>   |             |
| <code>\$.!Internet.files.startup</code>   | <code>startup</code>    |             |

The `hosts`, `networks`, `protocols` and `services` files are collectively known as the *databases*.

### What the files do

- The `!Configure` file does most of the configuration of the software. It sets the principal host name of a computer. It configures each interface, setting their Internet addresses, their netmasks, and the driver modules to be used. It defines where to find the other configuration files –

one location for the *databases*; and another location for the **startup** file. It specifies whether the RouteD module should be run to establish routing information. Finally, it sets whether or not the station will forward packets between multiple interfaces, and hence whether it acts as a TCP/IP gateway.

- The **hosts** file gives the host names and Internet addresses of all the computers you wish to refer to by their host name.
- The **networks**, **protocols** and **services** files contain databases of network, protocol and service names. These files are unused by the TCP/IP Protocol Suite (Release 2), and are provided to support any extra software that uses TCP/IP protocols.
- The files specify the default values normally used on all computers that support the TCP/IP protocols; consequently, you shouldn't ever need to edit them. If you do, you should see respectively the UNIX *networks* (5), *protocols* (5), and *services* (5) manual pages.
- The **startup** file initialises your computer's interface(s), and also establishes routes to remote networks or hosts – if you're not using RouteD to do so.

## Different ways to configure the software

There are some decisions you have to make on how to configure the software. This section outlines what those decisions are. For details of how to configure the choices you make, see Installing TCP/IP within OmniClient on page 56, and the instructions and examples in the files you'll need to edit.

### Ways to set the Internet address of each interface

You can set the Internet address of each interface in three ways:

- you can set it explicitly on the computer to which it's fitted
- you can use the interface name to look up the Internet address in the **hosts** database



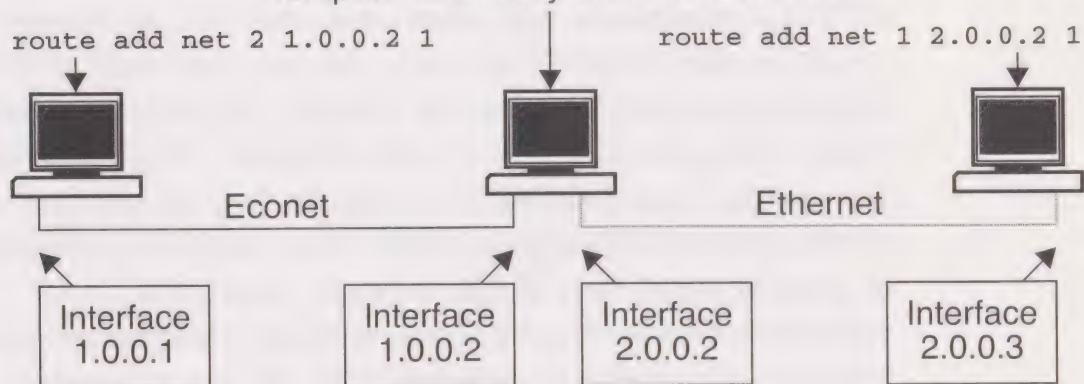
- you can use the physical address to look up the Internet address using *Reverse ARP* exchanges with an *ARP server* (see Setting up an ARP server, if required on page 60).

## Ways to establish routing information

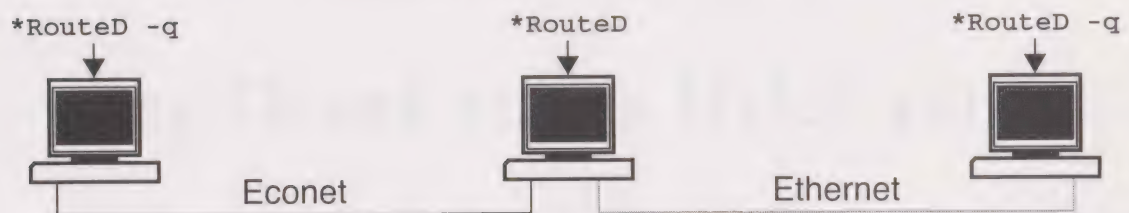
If you are using gateways, you need to ensure that each RISC OS computer knows their location, and the route to subnets other than the one to which it is connected. There are three ways you can do this:

- Use the **\*Route** command to explicitly define the routing of each gateway. For example:

No route command needed because computer is already on both networks

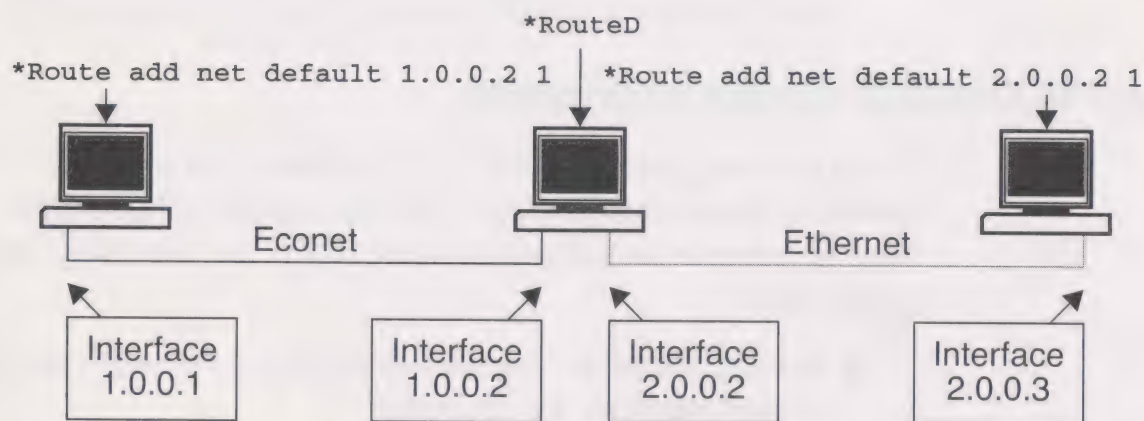


- Use the **Routed** module to perform the routing for you. For example:



- Use a hybrid of the above two methods. For stations with a single interface, use the **\*Route** command to define the location of a gateway. Use the **Routed** module on each gateway to establish the route to other networks and

gateways. For example:



The **Routed** module uses the Routing Information Protocol (or *RIP*) to communicate with other computers that implement *RIP* – such as other RISC OS gateways running **Routed**, or UNIX computers running the **routed** daemon – and hence to establish routes. This generates a lot of network traffic. We recommend that you only run **Routed** if you have a very complex network on which other computers use *RIP*. There are no real advantages to using **Routed** on a simple network, where it degrades performance unnecessarily; we recommend that you instead use **\*Route** commands, as outlined in the first example above.

You can get more background about the **\*Route** command and the **Routed** module from page 90 onwards of the *TCP/IP Protocol Suite (Release 2) User Guide*.

## Installing TCP/IP within OmniClient

### Backing up the distribution discs

In setting up stations you'll need to alter the software. **You must not alter the distribution discs themselves**; always work on copies of them, made either on media that you can access from other computers (i.e. an Acorn file server) or that you can take to other computers (i.e. a floppy disc):

- If your RISC OS computers can access an Acorn file server, make a directory on the file server called (say)



**TCP\_IP**, and open its directory display.

Put the *Network* distribution disc in the floppy drive, open its directory display, select all the files and drag them to the **TCP\_IP** directory display. Dismount the floppy disc. Repeat this process for the *Applications* distribution disc.

- Alternatively, use the **Backup** option from the floppy disc drive's icon bar menu to back up the two distribution discs to another pair of floppy discs.

## Where you can install the software

When you actually install the software for a particular station, you can do so on any media you like, such as a hard disc, floppy discs or a remote file server. All that is important is that you preserve the directory structure that is on the distribution discs – things that are in the same directory must stay together.

- If you're installing the software on a hard disc or an Acorn file server, we suggest you make a directory called (say) **TCP\_IP** to hold the software. This directory can be anywhere you like – it needn't be in the root directory.
- You don't have to do this, and may prefer to put the software in a directory that already contains other applications.

## Configuring the software

- 1 Load Edit onto the icon bar – if it's not already loaded.
- 2 Open the **!Internet** application directory by holding down the Shift key while you double-click on its icon. (**!Internet** is on the *Network* distribution disc.)
- 3 If you have a third-party network interface fitted that has a disc-based driver, add that driver to the **drivers** subdirectory:
  - Open the directory display that shows the third-party driver.
  - Open the **drivers** subdirectory of the **!Internet** application by double-clicking on its icon.

- Drag the third-party driver from its directory display to the **!Internet.drivers** directory.

Take note of the name of the driver; you'll need to know it for the next step.

#### 4 Edit the **!Configure** file:

- Load it into Edit by dragging its icon to the Edit icon on the icon bar.
- Following the instructions in the file, edit the lines that set the system variables:

| Variable                 | Notes  |
|--------------------------|--|
| <b>Inet\$HostName</b>    | This sets a station's principal host name.   |
| <b>Inet\$EcoIPAddr</b>   | This sets whether your station has an Econet interface that you wish to configure for use with TCP/IP, and (if so) sets its Internet address, or specifies that it be looked up using Reverse ARP.   |
| <b>Inet\$EcoIPMask</b>   | This sets the netmask for an Econet interface; the default is correct for most cases.  |
| <b>Inet\$EtherIPAddr</b> | This sets whether your station has an Ethernet interface that you wish to configure for use with TCP/IP, and (if so) sets its Internet address, or specifies that it be looked up using Reverse ARP. |
| <b>Inet\$EtherIPMask</b> | This sets the netmask for an Ethernet interface; the default is correct for most cases.  |
| <b>Inet\$EtherDevice</b> | This sets the driver module to be used to interface the TCP/IP software with your Ethernet interface.  |
| <b>InetDBase\$Path</b>   | This sets the pathname of the directory containing the database files.   |
| <b>Inet\$Startup</b>     | This sets the pathname of the <code>startup</code> file.   |



| Variable                  | Notes  |
|---------------------------|--|
| <b>Inet\$RouteOptions</b> | This sets whether RouteD – the Internet routing module – is run, and (if so) with what options.  |
| <b>Inet\$IsGateway</b>    | This sets whether the machine is to forward IP packets, and hence whether it will act as a gateway. <ul style="list-style-type: none"> <li>● Save the edited <b>!Configure</b> file, overwriting the old version.</li> </ul> |

5 Open the **files** directory.

6 Edit the **hosts** file:

- Load it into Edit by dragging its icon to the Edit icon on the icon bar.
- Edit the file so that it contains the Internet addresses and host names of all the RISC OS computers you wish to refer to by host name. (If a computer's host name isn't in here you'll instead have to use its Internet address whenever you want to communicate with it.)
- If you want to add any UNIX hosts to the file, their addresses and names must match those in your UNIX **/etc/hosts** files. For a small number of hosts you'll probably find it best just to print out and copy a UNIX **hosts** file. For a larger number of hosts, you might find it better to copy across your UNIX **hosts** file later.

If you need help, see the comments in the supplied **hosts** file, and the UNIX *hosts* (5) manual page.

Then save the edited **hosts** file:

- If you did not change the value of **InetDBase\$Path** in the **!Configure** file, just overwrite the file you loaded.
- Otherwise, save it to the new location you set up when you edited the **!Configure** file. Copy the other databases (the **networks**, **protocols**, and **services** files) to the same new location.

Delete the old databases from the **files** directory.

- 7 If you are using gateways but are not using RouteD, edit the **startup** file:
  - Load it into Edit by dragging its icon to the Edit icon on the icon bar.
  - Add **\*Route** commands to the file so that it contains details of all the gateways you will use.

For guidance and examples see the comments in the file, and the documentation of the **\*Route** command on page 90 of the *TCP/IP Protocol Suite (Release 2) User Guide*.

Then save the edited **startup** file:

- If you did not change the value of **Inet\$Startup** in the **!Configure** file, just overwrite the file you loaded.
  - Otherwise, save it to the new location you set up when you edited the **!Configure** file. Delete the old version of the file from the **files** directory. If there's nothing left in the **files** directory, delete it too.
- 8 Finally, remove write permission from the files so that your work won't be undone. If you've been using floppy discs, you may also like to write protect them.

## Setting up an ARP server, if required

If you want to use Reverse ARP exchanges to map physical addresses to Internet addresses you'll need to set up an ARP server. The machine doing this can be a RISC OS or a UNIX computer, and can do so over either Ethernet or Econet.

If you've already got a UNIX ARP server running, the most sensible thing to do is to add entries for your RISC OS machines to the database it uses.

If you need to set up a RISC OS ARP server it must *publish* its entries. Use the command:

```
*ARP -f filename
```



in a boot file.

You can find the physical address(es) of a RISC OS computer's interface(s) using the relevant **\*EnInfo** command; see page 81 of the *TCP/IP Protocol Suite (Release 2) User Guide* for further details. You may find it helpful to know that the physical address of an Econet interface is:

`00.00.00.00.station_number.net_number`

For more details, see the documentation of the **\*ARP** command on page 78 of the *TCP/IP Protocol Suite (Release 2) User Guide*.

You've now finished installing the TCP/IP Protocol Suite (Release 2) on RISC OS.

## Advanced installation

If you have a good knowledge of RISC OS system variables, you will be able to see the wide range of ways you can set up the TCP/IP Protocol Suite (Release 2). In the above text, and in the comments in the configuration files, we've laid out several different ways which should suit most possible installations – but you may be able to see a way that is better suited to your site. If you do, and if you know what you're doing, by all means further adapt the configuration files. However, you should ensure that you keep, unchanged, a backup copy of the distribution discs.





# 10 Using !Internet

This chapter tells you how to set an IP address. It also gives you information about a variety of modules and \* Commands that duplicate UNIX commands. If you're a more experienced UNIX or !Internet user you may well find these useful.

## Setting an IP address

The Internet application allows you to communicate with other computers on a fully-configured TCP/IP network. For this to work, you need to edit some of the files in the !Internet application to reflect the state of your network.

### Editing the Configure file

Firstly, you need to edit the file **!Internet.!Configure** so that it specifies each machine (host) on the network.

Each host specified in the Configure file must have a unique IP address: give each host a unique name, and then translate this name to a number via the hosts file (see below). The Configure file is fully-commented, so there's no need to describe it in detail here.

### Editing the hosts file

You'll also need to alter the hosts file, **!Internet.files.hosts**. This forms the link between the host name and the IP address (see page 49 for details of obtaining unique IP addresses). The file has the format:

```
127.0.0.1 loopback localhost loghost
1.0.0.1      ahost
2.0.0.1      anotherhost
2.0.0.2      agateway
3.0.0.1      remotehost
```

You need to specify computers in this list to which software may

need to refer to the remote computer by name. This may include server and gateway machines. Do **not** delete the loop-back entry.

## Running !Internet

To use any of these commands you need to run the Internet application. There are three ways you can do this:

- double-clicking on the Internet icon from the desktop
- typing a command at the command line
- including a command in a boot file.

In both the latter cases you must use a command of the form:

**\*Run Internet\_pathname.!Internet.!Run**

where *Internet\_pathname* is the rest of the pathname to the Internet application. You **must** start the command with **\*Run**; if you don't RISC OS won't know which filing system holds the Internet application.

### Getting a command line

If you're using the desktop, there are three ways of getting a command line so you can enter **\* Commands**:

- Open a task window by pressing Ctrl-F12, or by choosing **Task window** from the Task Manager's icon bar menu.
- Temporarily leave the desktop by pressing F12, or by choosing **\* Commands** from the Task Manager's icon bar menu.
- Permanently leave the desktop by choosing **Exit** from the Task Manager's icon bar menu.

## Internet module \* Commands

The core of the Internet application is a module named



**Internet.** It provides two \* Commands:

| Command             | Use  | Page |
|---------------------|--|------|
| <b>*InetGateway</b> | Toggle IP packet forwarding.                 | 77   |
| <b>*InetInfo</b>    | Display Internet module internal statistics. | 78   |

Running the Internet application loads the Internet module; you can then use the above \* Commands.

## Absolute programs

There are also a set of commands that are provided as Absolute files (that is, each file is a program that performs one command) in the **bin** subdirectory of the Internet application. These are:

| Command           | Use  | Page |
|-------------------|--|------|
| <b>*ARP</b>       | Address resolution display and control.                      | 68   |
| <b>*IfConfig</b>  | Configure network interface parameters.                      | 73   |
| <b>*IfRConfig</b> | Configure network interface parameters from a remote server. | 76   |
| <b>*Ping</b>      | Send ICMP ECHO_REQUEST packets to network hosts.             | 79   |
| <b>*Route</b>     | Manually manipulate the routing tables.                      | 81   |

Running the Internet application adds the **bin** subdirectory to the system variable **Run\$Path**. RISC OS then knows where to find the above \* Commands, so you can use them directly from the command line.

Although the above are actually programs, in the section that follows we've treated them as \* Commands because you're most likely to use them in just the same way.

## Ethernet driver module \* Commands

The Ethernet driver modules (held in the **drivers** subdirectory of the application) each provide a single \* Command to give information.

Ether3 refers to the A5000 style modules and EtherB to the Risc PC network card. Cards produced by different manufacturers may have differences – such as EtherH (i-cubed) – but all are suitable.

The **Ether3** module also provides a configuration command:

| Command           | Use   | Page |
|-------------------|---|------|
| *Configure Ether3 | Sets the configured state of  | 70   |
| *Configure EtherB | Acorn Ethernet 3 cards.   |      |
| *EnInfo           | Display details of physical interface activity, including their physical addresses. | 71   |

*Note* There is no \*Configure command for i-cubed (EtherH) cards.

Running the Internet application loads any appropriate Ethernet driver modules; you can then use any of the above \* Commands that are relevant to your interface(s).

## RouteD \* Commands

Finally, the Internet application also contains the **RouteD** module in its **rm** subdirectory. It provides three \* Commands:

| Command         | Use                                    | Page |
|-----------------|--|------|
| *RouteD         | Start the RouteD module.               | 83   |
| *RouteDTraceOff | Turn off tracing by the RouteD module. | 87   |
| *RouteDTraceOn  | Turn on tracing by the RouteD module.  | 88   |

To use these \* Commands you need to load the RouteD module. By default the Internet application doesn't do so when you run it – but you can configure it so it does do so. See the *TCP/IP*



*Protocol Suite (Release 2) Installation Guide* for details, or ask your system administrator.

## \*ARP

Address resolution display and control.

### Syntax

```
*ARP host
*ARP -a
*ARP -d host
*ARP -s host phys_addr [temp] [pub]
*ARP -f filename
```

### Parameters

|                  |  |
|------------------|--|
| <i>host</i>      | An Internet host specified either by name (which must be present in the host name data base <code>&lt;InetDBase\$Path&gt;hosts</code> ) or by address (using the standard Internet dot notation) |
| <i>phys_addr</i> | The physical address of <i>host</i> given in Ethernet format (i.e. six hexadecimal bytes separated by colons)  |
| <i>filename</i>  | The full pathname of a file containing multiple entries to be set in the ARP table   |

### Use

The ARP program displays and modifies the Internet-to-Physical-address translation tables located in the Internet module and used by the address resolution protocol ARP.

With no flags, the program displays the current ARP entry for *host*.

The **-a** flag makes the program display all the ARP entries currently in its table.

The **-d** flag makes the program delete an entry for *host*.

The **-s** flag makes the program create an ARP entry for the host called *host* with the physical address *phys\_addr*.

- The entry will be permanent unless the word **temp** is given in the command.



- The entry will be 'published' if the word **pub** is given. This system will then act as an ARP or Reverse ARP server, responding to requests for *host*'s physical address even though the host address is not its own.

The **-f** flag causes the file *filename* to be read and multiple entries to be set in the ARP tables. Entries in the file should be of the form:

```
host phys_addr [temp] [pub]
```

with argument meanings as given above.

If you don't know the physical address of an interface, you can use the \*InetInfo command to find it.

For your reference, the physical address of an Econet interface takes the form:

```
00.00.00.00.station_number.net_number
```

## Examples

```
*ARP tp1
*ARP -a
*ARP -d 01.01.01.01
*ARP -s tp1 01.01.01.01.01.01 temp
*ARP -f adfs::HardDisc.$.Internet.ARP_Table
```

**Related**           None  
**commands**

## \*Configure Ether3 EtherB

## Sets the configured state of Acorn Ethernet 3 cards.

## Syntax

```
*Configure Ether3 Enable|Disable|Default|Terse|
Verbose [card]
```

|                   |             |   |
|-------------------|-------------|---|
| <b>Parameters</b> | <i>card</i> | The expansion card's number, as given by *Podules |
|-------------------|-------------|---|

**Use** \*Configure Ether3 sets the configured state of all fitted Acorn Ethernet 3 cards, or of a single *card* specified by its expansion card slot number.

\*Configure Ether3 NewInet enables a major optimisation present in newer versions of the Internet module, such as those used with this product and with AUN. This is the default. \*Configure Ether3 OldInet makes the card compatible with older versions of the Internet module, and should not be used with this product.

\*Configure Ether3 Enable permits the interface to be used – which is the default – whereas \*Configure Ether3 Disable prevents the interface from being used.

\*Configure Ether3 Default sets the configured state to the default: namely, NewInet and Enabled.

### Example

**Related commands** \*Status Ether3 (page 89)



Display details of physical interface activity, including their physical addresses.

## **Syntax**

|                |   |
|----------------|---|
| <b>*EnInfo</b> | Display details for all Acorn Ethernet 2 cards          |
| <b>*E3Info</b> | Display details for all ANT/Acorn Ethernet 3 cards      |
| <b>*EBInfo</b> | Display details for all ANT/Acorn Network slot cards    |
| <b>*EHInfo</b> | Display details for all i-cubed (I <sup>3</sup> ) cards |

## **Use**

\*EnInfo displays details of physical interface activity, including the physical addresses of all Ethernet interfaces of the relevant type that are fitted to the computer.

Most of the information displayed is difficult to understand. It is presented mainly as an aid to trouble-shooting, should you require it, for example to assist with technical support queries.

If you are using an interface that is not listed above, you should consult its documentation to see if its manufacturer has provided a similar command; we expect most will do so.

## **Example**

See next page.

## **Related commands**

\*InetInfo (page 78)

## Example

### \*E3Info

Ether3 interface statistics

ea0: 8005 16 bit MEMC1a, slot 2, up, hardware address 00:02:07:00:A3:48

Interface driver : ea  
Interface unit : 0  
Interface location : Expansion slot 2  
Interface address : 00:02:07:00:A3:48  
Interface controller: 8005 16 bit MEMC1a  
Initialise time : Wed Mar 1 11:32:02 1995  
Running time : 2 minutes, 5 seconds.  
Packets sent : 4362  
Packets received : 4830  
Bytes sent : 2276647  
Bytes received : 2352828  
Send errors :  
Receive errors :  
Broadcasts sent :  
Broadcasts received : 298  
Multicasts sent :  
Multicasts received : 111  
Monitor sends :  
Monitor receives :  
Send interrupts : 4362  
Receive interrupts : 4830  
Delivered packets : 4835  
Undelivered packets :  
Sends too large :  
Send max collisions :  
Send overflows :  
Sends blocked :  
Receives too small :  
Receives too large :  
Incomplete receives :  
Receive CRC errors :  
Lack of mbufs :  
Receive overflows :

Standard clients:

Frame = &0800, ErrLvl=00, AddrLvl=01  
Frame = &0806, ErrLvl=00, AddrLvl=01  
Frame = &8035, ErrLvl=00, AddrLvl=01

IEEE 802.3 client:

Frame = &0000, ErrLvl=00, AddrLvl=02

Log: Ether3 messages can appear here



## \*IfConfig

Configure network interface parameters.

**Syntax** `*IfConfig [-e] interface [inet host  
[parameters]]`

|                                    |  |
|------------------------------------|--|
| <b>Parameters</b> <i>interface</i> | The two character name (as defined by the manufacturer) and unit number (starting from 0 for the first of that type fitted) of an interface – e.g. <b>en0</b> (Acorn Ethernet 2 unit 0), <b>ea1</b> (Acorn Ethernet 3 unit 1), <b>ec0</b> (Econet) |
|------------------------------------|--|

|             |  |
|-------------|--|
| <i>host</i> | An Internet host specified either by name (which must be present in the host name data base <code>&lt;InetDBase\$Path&gt;hosts</code> ) or by address (using the standard Internet dot notation) |
|-------------|--|

## Use

The IfConfig program is used to assign an Internet address to a network interface supported by the Internet module, and/or to configure network interface parameters. It must be used at Internet module start-up time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters.

You can set the following parameters with the IfConfig program:

**up** Mark an interface 'up'. This may be used to enable an interface after going **\*IfConfig ... down**. It happens automatically when setting the first address on an interface. If the interface was reset when previously marked down, the hardware will be re-initialised.

|                            |   |
|----------------------------|---|
| <b>down</b>                | Mark an interface 'down'. When an interface is marked 'down', the system will not attempt to transmit messages through that interface. If possible, the interface will be reset to disable reception as well. This action does not automatically disable routes using the interface.  |
| <b>arp</b>                 | Enable the use of the Address Resolution Protocol in mapping between network level addresses and link level addresses (default), for mapping between Internet addresses and Ethernet addresses.   |
| <b>-arp</b>                | Disable the use of the Address Resolution Protocol.   |
| <b>metric <i>n</i></b>     | Set the routing metric of the interface to <i>n</i> (default 0). The routing metric is used by the routing protocol (Routed module). Higher metrics have the effect of making a route less favourable; metrics are counted as additional hops to the destination network or host.   |
| <b>netmask <i>mask</i></b> | Specify how much of the address to reserve for subdividing networks into sub-networks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number with a leading <b>0x</b> , as an Internet address using the standard dot notation, or as a pseudo-network name listed in the network table <b>&lt;Inet\$DBase&gt;networks</b> . The mask contains 1's for the bit positions in the 32-bit address which |



are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion.

**broadcast**

Specify the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all 1's.

If you don't use any of the optional parameters, the IfConfig program displays the current configuration for a network interface.

The **-e** option makes IfConfig place any fatal error report string in the system variable **Inet\$Error** instead of writing it to the standard output.

**Example**

```
*IfConfig -e ea0 inet host1 up
```

**Related  
commands**

\*IfRConfig (page 76)

# \*IfRConfig

Configure network interface parameters from a remote server.

**Syntax**      **\*IfRConfig** [-e] *interface* [revarp]  
                 [netmask]

**Parameters**    *interface*                      The two character name (as defined by the manufacturer) and unit number (starting from 0 for the first of that type fitted) of an interface – e.g. **en0** (Acorn Ethernet 2 unit 0), **ea1** (Acorn Ethernet 3 unit 1), **ec0** (Econet)

**Use**                      The IfRConfig program is used to assign an Internet address to a physical network interface. It does so using a Reverse ARP broadcast/response transaction with a remote server host, and/or (optionally) a netmask via an ICMP MASKREQ broadcast/response transaction. If it does not find a remote server, it generates an error.

The optional parameter **revarp** requests an interface address; **netmask** requests a netmask.

If you don't use any of the optional parameters, the IfRConfig program displays the current configuration for a network interface.

The **-e** option makes IfRConfig place any fatal error report string in the system variable **Inet\$Error** instead of writing it to the standard output.

**Example**                **\*IfRConfig ea0 revarp**

**Related commands**    **\*IfConfig** (page 73)



# **\*InetGateway**

Toggle IP packet forwarding.

**Syntax** `*InetGateway 1|0`

**Use** `*InetGateway` may be used to enable (`*InetGateway 1`) or to disable (`*InetGateway 0`) IP layer packet forwarding (i.e. gateway operation) if multiple network interfaces are present. The default state is off.

**Example** `*InetGateway 1`

**Related commands** None

## **\*InetInfo**

Display Internet module internal statistics.

### **Syntax**

**\*InetInfo [r] [p]**

### **Use**

\*InetInfo displays detailed information about Internet module activity. By default it only gives details of internal resource usage and protocol activity using the options:

**r**                                      Display only internal resource information (the default)

**p**                                      Display only protocol information

Most of the information displayed is quite technical in nature. It is presented mainly as an aid to trouble-shooting, should you require it.

The **i** option used in Release 1 of the TCP/IP Protocol Suite to give information on an interface has been replaced by the \*EnInfo command.

### **Example**

**\*InetInfo rp**

### **Related commands**

\*EnInfo (page 71)



## \*Ping

Send ICMP ECHO\_REQUEST packets to network hosts.

**Syntax**      **\*Ping** [-r] [-v] *host* [*packetsize*] [*count*]

|                   |                   |   |
|-------------------|-------------------|---|
| <b>Parameters</b> | <i>host</i>       | An Internet host specified either by name (which must be present in the host name data base<br><InetDBase\$Path>hosts) or by address (using the standard Internet dot notation) |
|                   | <i>packetsize</i> | The size of packet to send (default is 64 bytes)  |
|                   | <i>count</i>      | The number of packets to send (default is 1)  |

**Use**      An Internet can be a large and complex aggregation of network hardware, connected together by gateways. Tracking a single-point hardware or software failure can be difficult. The Ping program utilises the ICMP protocol's mandatory ECHO\_REQUEST datagram to elicit an ICMP ECHO\_RESPONSE from a host or gateway. ECHO\_REQUEST datagrams (or *pings*) have an IP and ICMP header, and then an arbitrary number of padding bytes used to fill out the packet.

The optional **-r** parameter makes the Ping program bypass the normal routing tables and send directly to a host on an attached network. (If the host is not on a directly attached network an error is returned.) You can use this option to 'ping' a local host through an interface that has no route through it.

The optional **-v** parameter causes verbose output: any ICMP packets other than ECHO RESPONSE that are received are listed.

When using Ping for fault isolation you should first run it on your local host, to verify that your local network interface is up and running. Then you should 'ping' hosts and gateways that are further and further away.

If you don't give a *count* the Ping program sends just one datagram and expects a single ECHO\_RESPONSE to be returned. If you give a *count* Ping sends one datagram per second up to that number, and prints one line of output for every ECHO\_RESPONSE returned. No output is produced if there is no response.

Packet loss statistics are computed and a brief summary is displayed when all responses have been received, or when the program times out, or when you terminate the program.

RISC OS Ping messages are not time-stamped.

**Example**            `*Ping -v tp1 1024 20`

**Related  
commands**        None



Manually manipulate the routing tables.

## Syntax

```
*Route [-e] [-f] add [net|host]  
destination gateway metric
```

```
*Route [-e] [-f] delete [net|host]  
destination gateway
```

## Parameters

|                    |   |
|--------------------|---|
| <i>destination</i> | An Internet host or network specified either by name (which must be present in the respective host or network name data base <code>&lt;InetDBase\$Path&gt;...</code> ) or by address (using the standard Internet dot notation) |
| <i>gateway</i>     | The next-hop gateway to which packets should be addressed   |
| <i>metric</i>      | A count giving the number of hops to the destination  |

## Use

You can use the Route program to manually manipulate the Internet module's network routing tables.

The *metric* must be zero if the destination is on a directly-attached network, and non-zero if the route utilises one or more gateways. If you're adding a route with *metric* 0, the *gateway* given is the address of this host on the common network, indicating the interface to be used for transmission.

Routes to a particular host are distinguished from those to a network by interpreting the Internet address associated with *destination*. The optional keywords `net` and `host` force the *destination* to be interpreted as a network or a host, respectively. Otherwise, if the *destination* has a local address part of 0 or if it's the symbolic name of a network, then the route's presumed to be to a network; else the route's presumed to be to a host. All symbolic names specified for a destination or gateway are looked up first as a host name; if this

fails, the name is then looked up as a network name.

The **-e** option makes Route place any fatal error report string in the system variable **Inet\$Error** instead of writing it to the standard output.

The **-f** option makes Route delete all gateway entries in the Internet module's routing tables. If this is used in conjunction with an **add** command, the tables are flushed first.

## Examples

```
*Route -f add net 1 0.0.0.1 1
```

```
*Route -e delete host farhost gatehost
```

## Related commands

```
*RouteD (page 83)
```



Start the RouteD module.

## Syntax

**\*RouteD**

## Use

RouteD may be invoked to manage the network routing tables within the Internet module.

When RouteD is started, it finds those directly connected interfaces configured into the system and marked 'up'. If multiple interfaces are present, it is assumed that the host will forward packets between networks. RouteD then broadcasts a request packet on each interface and subsequently listens continuously for request and response packets from other hosts. Notification of the arrival of these packets is performed in the background, via RISC OS events.

When a request packet is received, RouteD formulates a reply based on the information maintained in its internal tables. The response packet generated contains a list of known routes, each marked with a hop count metric (a count of 16, or greater, is considered 'infinite'). The metric associated with each route returned provides a metric relative to the sender.

Response packets received by RouteD are used to update the routing tables if one of the following conditions is satisfied:

- No routing table entry exists for the destination network or host, and the metric indicates the destination is reachable (i.e. the hop count is not infinite).
- The source host of the packet is the same as the router in the existing routing table entry. That is, updated information is being received from the very internetwork router through which packets for the destination are being routed.
- The existing entry in the routing table has not been updated for some time (defined to be 90 seconds) and the route is at least as cost-effective as the current route.
- The new route describes a shorter route to the destination

than the one currently stored in the routing tables. (The metric of the new route is compared against the one stored in the table to decide this.)

When an update is applied, RouteD records the change in its internal tables and updates the Internet module routing table. The change is reflected in the next response packet sent.

In addition to processing incoming packets, RouteD also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed an additional 60 seconds to insure the invalidation is propagated throughout the local Internet.

Hosts acting as internetwork routers gratuitously broadcast their routing tables every 30 seconds to all directly connected networks. The normal routing tables are bypassed when sending gratuitous responses. The reception of responses on each network is used to determine that the network and interface are functioning correctly. If no response is received on an interface, another route may be chosen to route around the interface, or the route may be dropped if no alternative is available.

## **Inet\$RouteDOptions system variable**

RouteD supports several option flags, recorded in **Inet\$RouteDOptions**:

- |          |  |
|----------|--|
| <b>g</b> | This flag is used on internetwork routers to offer a route to the 'default' destination. This is typically used on a gateway to the Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers. |
| <b>s</b> | Supplying this option forces RouteD to supply routing information whether it is acting as an internetwork router or not. This is the default if multiple network interfaces are configured for TCP/IP.   |



|                |  |
|----------------|--|
| <b>q</b>       | This is the opposite of the <b>s</b> option. |
| <b>DEFAULT</b> | This option forces the defaults to be used.  |

You should set this system variable with a line having the syntax:

```
*Set Inet$RouteDOptions [g][s|q][DEFAULT]
```

## Passive and active gateways

In addition to the facilities described above, RouteD supports the notion of 'distant' passive and active gateways. When RouteD is started up, it reads the file `<InetDBase$Path>gateways` (if present) to find information about such gateways. Gateways specified in this manner should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange routing information (i.e. they should have a RouteD process running on the machine).

Passive gateways are maintained in the routing tables forever and information regarding their existence is included in any routing information transmitted.

Active gateways are treated equally to network interfaces. Routing information is distributed to the gateway and if no routing information is received for a period of time, the associated route is deleted.

External gateways are also passive, but are not placed in the routing table nor are they included in routing updates. The function of external entries is to inform RouteD that another routing process will install such a route, and that alternate routes to that destination should not be installed. Such entries are only required when both routers may learn of routes to the same destination.

## Gateways file format

The `<Inet$DBase>gateways` file is comprised of a series of lines, each in the following format:

```
net|host destination gateway gateway metric metric passive|active|external
```

- The `net` or `host` keyword indicates if the route is to a network or specific host.

- The ***destination*** parameter gives an Internet host or network specified either by name (which must be present in the respective host or network name data base **<InetDBase\$Path>...**) or by address (using the standard Internet dot notation).
- The ***gateway*** parameter gives the name or address of the next-hop gateway to which packets should be forwarded.
- The ***metric*** parameter is a count giving the number of hops to the destination.
- One of the keywords **passive**, **active** or **external** indicates if the gateway should be treated as passive or active (as described above), or whether the gateway is external to the scope of the RouteD protocol.

### Example

**\*RouteD**

### Related

### commands

\*Route (page 81) \*RouteDTraceOff (page 87)  
\*RouteDTraceOn (page 88)



## \*RouteDTraceOff

Turn off tracing by the RouteD module.

**Syntax**      \*RouteDTraceOff

**Use**            \*RouteDTraceOff turns off tracing by the RouteD module.

**Example**        \*RouteDTraceOff

**Related**        \*RouteD (page 83)

**commands**    \*RouteDTraceOn (page 88)

# **\*RouteDTraceOn**

Turn on tracing by the RouteD module.

|                         |   |  |
|-------------------------|---|--|
| <b>Syntax</b>           | <b>*RouteDTraceOn</b> [ <i>trace_level</i> ] [ <i>filename</i> ]  |  |
| <b>Parameters</b>       | <i>trace_level</i>  | Level (1-4) of trace required, where 1 is the lowest level and 4 the highest |
|                         | <i>filename</i>   | A valid pathname specifying the file to which to output trace                |
| <b>Use</b>              | <p>*RouteDTraceOn turns on tracing by the RouteD module. The output may be written either on the standard output or into the given logging file (which remains open until *RouteDTraceOff is invoked).</p> <p>Most of the information displayed is technical and difficult to interpret. It is presented mainly as an aid to trouble-shooting, should you require it.</p> |  |
| <b>Example</b>          | <b>*RouteDTraceOn 2 adfs::HardDisc.\$.RDTraceOut</b>  |  |
| <b>Related commands</b> | <b>*RouteD</b> (page 83)  |  |
|                         | <b>*RouteDTraceOff</b> (page 87)  |  |



# **\*Status Ether3 EtherB**

Displays the configured state of ANT/Acorn Ethernet 3 cards.

## **Syntax**

**\*Status Ether3**

## **Use**

\*Status Ether3 displays the configured state of all fitted ANT/Acorn Ethernet 3 cards.

## **Example**

```
*Status Ether3  
ether3 enabled 0  
ether3 newinet 0
```

## **Related commands**

\*Configure Ether3 (page 70)





# 11 File mapping

ONE problem with different types of computers communicating is that the different systems have different conventions for naming files, using special characters in file names, using file types and so on.

This means that the process of retrieving files is often a more complex process than it might be. OmniClient's methods for dealing with the problem are described in this chapter.

The problem principally affects the 'foreign' file systems NFS and LanMan/Windows.

The following terms are used within this chapter:

|                    |   |
|--------------------|---|
| <b>Client-name</b> | Name of OmniClient module.  |
| <b>Mapping</b>     | Definition of file type translation between filing systems.   |
| <b>Extension</b>   | Alien filing system extension string. This DOES NOT include the separator string (eg, '/' or '.') and may be wildcarded, with '?' for a single character and '*' for multiple characters. Matches are case-insensitive, unless set to case sensitive by use of an optional flag. Up to seven characters of 8-bit ASCII are allowed. |
| <b>Filetype</b>    | RISC OS numeric filetype, given as hex (0x000-0xFFFF) a special filetype of 0x9999 is taken to be the default to allow all other RISCOS filetypes to be mapped to a given extension string. &NNN is accepted as well as 0xNNN.  |
| <b>Flags</b>       | Optional flags for this mapping (see below).  |

## Using extensions

The Extensions file, which is located at **!Omni.Files.Extensions**, allows OmniClient to perform automatic translations of filename and file type information between RISC OS and alien filing systems in a user-defined

manner.

Its purpose is to enable files on remote filing systems which have non-RISC OS file type styles, eg putting .EXE on the end, to be translated into a form which RISC OS can act upon without further intervention.

Only the non-RISC OS filing systems are candidates for this kind of translation, so NetFS (AUN Level4) and ShareFS (Access/+) will not exploit this feature. Of the other two, NFS does not yet use this feature, but LanManFS does.

By default, !Omni is supplied with the file noted above containing a set of sensible file extension translations which are useful for LanManFS users (for example, people with DOS files as originals). Users can customise the file according to their own requirements by following these instructions.

## Extensions file format

The Extensions file consists of sequence of blocks of mappings for a given protocol module (ie filing system).

A mapping string consists of three components: filetype, file extension and flags. The first two may be used either way around, ie to map filetypes to extensions or vice versa. The flags are documented below.

An extension takes the form of up to seven characters of 8-bit ASCII excluding the separator (eg "."). This will typically be a three-character DOS-style extension.

A filetype takes the form **&xxx** where **xxx** are hexadecimal digits.

Flags are the lowercase characters **k**, **s** and **c** – see below.



The general syntax of the Extensions file is as follows:

```
# comment line

client-name          # NB no comma after a client-name

extension,filetype,flags  # .extension to RISCOS file type mappings
extension,filetype,flags
...
filetype,extension,flags  # RISC OS filetype to .extension mappings
filetype,extension,flags
...
```

Here is an example from the Extensions file shipped with !Omni.  
It introduces the default set of file mappings:

```
---begin---

default             # Default file extension mappings
arc,&ddc             # SparkFS compressed file types
zoo,&ddc
zip,&ddc
...
&694,mac            # Reverse mappings
&695,gif
---ends---
```

Possible client names are the same as the short names for filing systems used in the Mounts and Startup files, so the following are valid:

|                |   |
|----------------|---|
| <b>default</b> | Used for all filing systems unless overridden |
| <b>LanMan</b>  | For LanManFS                                  |

There is always a block with a client-name of 'default'. This provides default mappings that the user can override by having a mapping entry for a specific filing system in its own block.

The mapping strings are prioritised using the order in which they appear in the file, ie, the higher up in the file they appear, the

higher the priority. Hence a \* wildcard will match everything if it appears first in the file (which is not what you want!)

Comments can appear anywhere in the file (started by a # character), and last to the end of the line. A new block of mappings for a different client module is marked by a valid client name with no extra punctuation (so that it cannot be confused with a file extension string).

If the client name is not registered at the time the file is parsed, mappings following are skipped. This has the result that if the filetype mappings file is parsed before client modules are loaded, then no mappings can take place. The solution to this problem that OmniClient uses is that every time a client module registers, the filetypes mapping file is reparsed. However, this normally only occurs once during startup – it is only if client modules register at some later time that the file is reparsed.

## Flags

The flags string is optional for every mapping. It consists of a sequence of single characters, with no separators, that each relate to a specific feature to turn on. The currently defined flags are;

| Flag char | Meaning  |
|-----------|--|
| k         | Keeps file extensions when Alien->RISC OS mapping. For example, <b>READ.ME</b> becomes <b>READ/ME</b> (type 0xFFFF). The default action is to remove these extensions.   |
| s         | Creates a subdirectory when Alien->RISC OS mapping. For example, <b>PROG.C</b> can become <b>C.PROG</b> (type 0xFFFF). Note that support for this flag is optional, and Client writers may choose not to implement it. |
| c         | Makes extension matches case sensitive (the default is case-insensitive). This would allow <b>.z</b> (gzip) and <b>.Z</b> (compress) file extensions to be distinguished, for example.                                 |



# NFS file mapping

There are a number of differences between the UNIX and RISC OS models of a filing system, the more important being:

- length of filenames
- use of special characters in filenames
- numbers of attribute bits stored with files
- meaning of attribute bits
- use of file types
- soft links.

Because of these clashes changes have to be made when mapping RISC OS file names and attributes to UNIX ones, and vice versa. Generally the changes made when mapping one way are reversed when mapping the other way, so the system is as transparent as possible if only viewed from RISC OS. If you view the files using the remote filestore though, you'll notice some differences.

This chapter outlines how the mapping of file names takes place, and what differences you'll notice between the RISC OS view of a file and the UNIX view.

## File mapping from RISC OS to UNIX

This section describes how RISC OS NFS maps files from RISC OS to UNIX.

### Filenames

#### Character translation

The first change RISC OS NFS makes to a filename is to translate the character '/' (the UNIX directory separator) to '.', for example:

| RISC OS name | UNIX name |
|--------------|-----------|
| fred/c       | fred.c    |
| /profile     | .profile  |

## File type extensions

RISC OS NFS then adds a filename extension to store the RISC OS file type.

You can set the extension used for any given file type to one of your choice. To do so you must edit the **extensions** file, held within the Internet application. See *Editing the extensions file* on page 103.

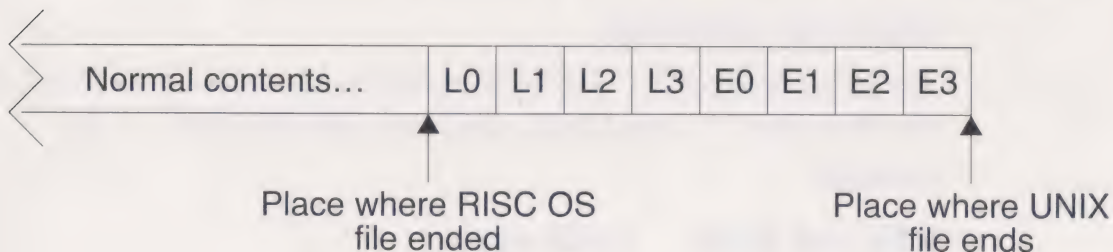
If you haven't set up a filename extension for a given file type, then a default extension is used instead. The default mapping of a RISC OS file called **Fred** is as follows:

| RISC OS type | UNIX name | Notes                             |
|--------------|-----------|-----------------------------------|
| Text         | Fred      |                                   |
| UNIX Ex      | Fred      | UNIX 'X' mode bit set             |
| Draw (&AFF)  | Fred,aff  |                                   |
| Obey (&FEB)  | Fred,feb  | and other file types similarly... |
| dead         | Fred,xxx  |                                   |
| untyped      | Fred,lxa  |                                   |
| directory    | Fred      |                                   |

A dead file is one that has been created but the contents of which are being updated. For example when NetFS copies a file to a file server it reserves space by creating a dead file before writing to it.

## File contents

The contents of files are unchanged when transferring to UNIX, save for untyped files. These have their load and execute addresses appended to the file, making it 8 bytes longer:



L0 is the least significant byte of the load address, L3 the most



significant. Bytes E0 to E3 are the execute address.

## Access attributes

### When creating a new file or directory

You can use the system variable **NFS\$CreateAccess** to define the default read/write access attributes for user, group and other that RISC OS NFS sets when creating a file or directory on UNIX. This variable uses six of its lowest nine bits:

| User |   |   | Group |   |   | Other |   |   |
|------|---|---|-------|---|---|-------|---|---|
| r    | w | - | r     | w | - | r     | w | - |

You can set it in octal by using a leading '0' (you'll find this familiar if you've ever used the UNIX `chmod` command with numbers), or in hexadecimal by using a leading '0x', or in decimal by just using a number. So the following would all set the variable to specify user read/write access, group read only access, and no access to others:

```
*Set NFS$CreateAccess 0640      (using octal)
*Set NFS$CreateAccess 0x1A0      (using hexadecimal)
*Set NFS$CreateAccess 416        (using decimal)
```

You can override the value of the **NFS\$CreateAccess** variable for a specific mount by setting a system variable **NFS\$CreateAccess\_mountname**.

You should set these access variables in a boot file; see your *RISC OS 3 User Guide* if you need help on this.

If a relevant access variable exists then files and directories are created with the read/write access it specifies. Files of type UNIX Ex also have their execute attributes set to be the same as the corresponding read bits in the variable.

If no relevant access variable exists then files are created with user read/write access, and with user execute permission if the files' type is UNIX Ex. Directories are created with user read, write and execute permission.

### When mapped from RISC OS

When RISC OS NFS sets the access to a UNIX file using RISC OS attributes they are mapped as follows:

| <b>RISC OS bit</b> | <b>UNIX bit</b>  |
|--------------------|--|
| owner read         | user read<br>user execute is also set if owner read is set and the file's type is UNIX Ex  |
| owner write        | user write   |
| public read        | group read and other read<br>group execute and other execute are also both set if public read is set and the file's type is UNIX Ex. |
| public write       | group write and other write  |
| locked             | (discarded)  |

Similarly, when RISC OS NFS sets the access to a UNIX directory using RISC OS attributes they are mapped as follows:

| <b>RISC OS bit</b> | <b>UNIX bit</b>   |
|--------------------|---|
| owner read         | ignored – i.e. user read is left unchanged                                |
| owner write        | ignored – i.e. user write is left unchanged<br>user execute is always set |
| public read        | group read and other read   |
| public write       | group write and other read  |
| locked             | NOT group execute and NOT other execute                                   |

### Dates

UNIX date stamps any files just as usual if you use RISC OS NFS to create or amend them.

### Finding an object

When RISC OS NFS is finding an object it searches in this order, using the first match it makes:

- 1 It searches for an exact name match.
- 2 It searches for an exact name match after any RISC OS specific extension has first been removed.
- 3 It searches for a name match ignoring case, after any



RISC OS specific extension has first been removed.

## File mapping from UNIX to RISC OS

This section describes how RISC OS NFS maps files from UNIX to RISC OS.

### Filename

#### *File type extensions*

The first change RISC OS NFS makes is to remove any filename extension used to store the RISC OS file type.

It starts by looking through the `extensions` file to see if the filename has an extension that matches one you specified; if so, the extension gets removed. You can in fact set up a different mapping for each direction of file transfer, so you can map many UNIX file extensions to single RISC OS file types. See section on page 103.

If RISC OS NFS can't find a matching filename extension in the `extensions` file it then tries to remove any of its own default extensions; so the following all appear as **Fred** under RISC OS:

| UNIX name | Notes                              |
|-----------|------------------------------------|
| Fred      |                                    |
| Fred,hhh  | hhh is three lower-case hex digits |
| Fred,xxx  |                                    |
| Fred,lxa  |                                    |

#### *Truncation*

The next thing RISC OS NFS does to a filename is to truncate it to the length set by the system variable

`NFS$TruncateLength`. By default this is set to the value 10 – the same length as the maximum that the desktop Filers can handle. It only gets read once, when the NFS module is loaded.

If you want a different truncate length use the `*Set` command, say in a boot file:

### **\*Set NFS\$TruncateLength 12**

If you're using NFS from the command line you may want to override filename truncation. To do so set the variable to a large number, e.g. 1000000.

## ***Character translation***

The final change RISC OS NFS makes to a filename is to translate the character '.' (the Acorn directory separator) to '/', for example:

| <b>UNIX name</b> | <b>RISC OS name</b> |
|------------------|---------------------|
| fred.c           | fred/c              |
| .profile         | /profile            |

## **File contents**

RISC OS NFS makes the last 8 bytes of any file with a '**,1xa**' extension invisible; this is to hide the load and execute addresses it presumes itself to have appended.

If you generate a file in UNIX with a '**,1xa**' extension which is less than 8 bytes long, you will get unpredictable behaviour if you try to manipulate it from RISC OS.

## **Access attributes**

When the access attributes of a UNIX file or directory get translated by RISC OS NFS they are mapped as follows:

| <b>UNIX bit</b> | <b>RISC OS bit</b>         |
|-----------------|----------------------------|
| user read       | owner read                 |
| user write      | owner write                |
| user execute    | (discarded)                |
| group read      | (discarded)                |
| group write     | (discarded)                |
| group execute   | (discarded)                |
| other read      | public read                |
| other write     | public write               |
| other execute   | (discarded for files)      |
|                 | NOT locked for directories |



## Dates

RISC OS NFS always uses the UNIX last modified date stamp to map to a RISC OS date stamp. It assumes the UNIX date stamp to be in GMT, and uses the value set by **\*TimeOffset** to convert this to local time. For details of **\*TimeOffset** see page 112.

## File types

RISC OS NFS resolves file types by looking for any filename extension used to store the RISC OS file type. It does so at the same time as it resolves filenames – see also the earlier section *Filenames* on page 99.

It starts by looking through the **extensions** file to see if the filename has an extension that matches one you specified; if so, it sets the file to the corresponding file type. See *Editing the extensions file* on page 103.

If RISC OS NFS can't find a matching filename extension in the **extensions** file it then sets the file type using its default file extensions. So, again taking the example of a file that will be displayed as **Fred**:

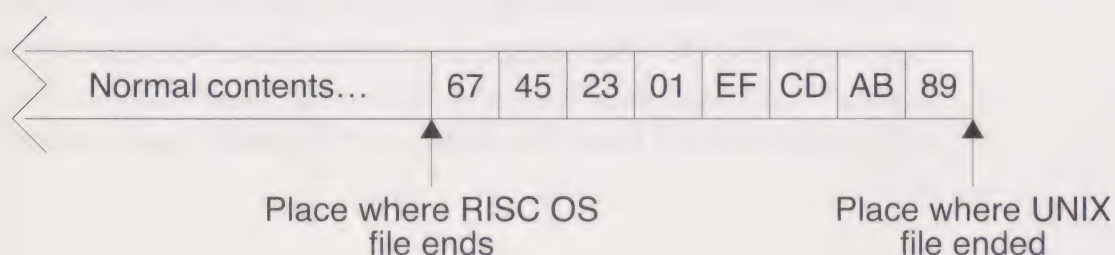
| UNIX name | Notes                          | RISC OS type     |
|-----------|--------------------------------|------------------|
| Fred      | UNIX directory                 | Directory        |
| Fred      | no execute bit is set          | Text             |
| Fred      | any execute bit is set         | UNIX Ex          |
| Fred,hhh  | hhh is 3 lower case hex digits | &hhh             |
| Fred,xxx  |                                | dead             |
| Fred,lxa  |                                | none,<br>undated |

See also *Soft links* on page 102.

## Load and execute addresses

If a UNIX file has the extension **,1xa** then RISC OS NFS assumes it to be a RISC OS untyped file that it created on UNIX. It uses the last 8 bytes of the file to give the load and execute

addresses. So if they were:



the load address would be &01234567, and the execute address would be &89ABCDEF.

## Soft links

RISC OS NFS resolves soft links up to eight times – that is, whilst following a soft link, it only allows eight soft links to be traversed. If this traversal reaches an existing object other than a soft link:

- the object's UNIX attributes and contents get used
- the soft link's UNIX name gets used to determine the RISC OS file type.

In other words soft links behave transparently except that, where there is more than one soft link to a file, its type may differ depending on which soft link you use to view it.

RISC OS NFS can't traverse a soft link that leaves a mount. If a UNIX link name starts with the character '/' then RISC OS NFS treats it as the root of its mount. Consequently absolute soft links will only work if you've mounted the UNIX root directory '/' and if the soft link does not leave the root filing system. For example, if you had mounted **/usr** then this UNIX soft link in the **/usr** directory would be traversed:

```
lrwxrwxrwx 1 root wheel 11 Feb 23 17:19 man -> ./share/man
```

whereas this one wouldn't be:

```
lrwxrwxrwx 1 root wheel 11 Feb 23 17:19 man -> /usr/share/man
```

We advise that when you make soft links on UNIX you always make relative links (that is, start them with '.' or '..') rather



than absolute ones.

If a soft link does not resolve to an existing non-soft-link object within eight expansions it's displayed as a file with type 'SoftLink' (&FDC). You can't do anything from RISC OS with one of these dead soft links.

## Other object types

Block and character special files and named sockets are displayed as UNIX Ex files. Fiddle with these from RISC OS at your peril!

## Editing the extensions file

The **extensions** file is held in the **files** subdirectory of the Internet application, and configures the mapping of RISC OS file types to UNIX filename extensions. To add your own filename extensions for specific RISC OS file types you need to edit this file:

- 1 Load Edit onto the icon bar – if it's not already loaded.
- 2 Open the directory display containing the **!Internet** application – if it's not already open.
- 3 Open the **!Internet** application directory by holding down the Shift key while you double-click on the **!Internet** icon.
- 4 Open the **files** directory.
- 5 Load it into Edit by dragging its icon to the Edit icon on the icon bar.
- 6 Add to the file your own mappings of file type to UNIX file extension.
  - There are two sets of mappings: one for files coming from RISC OS (starting immediately beneath the 'From extensions:' line), another for files returning to RISC OS (starting immediately beneath the 'To extensions:' line).
  - The general syntax is:

*RISC\_OS\_file\_type new\_extension*  
[*anything*]

The RISC OS file type can be the name of a file type, or its file type number in hexadecimal. So to give Data files (type &FFD) the extension '**.dat**' you could use either of these lines:

```
Data .dat  
ffd .dat
```

- If you add a third field ('**anything**' above) then the extension becomes 'sticky'.

When moving to UNIX the extension is only added if it's not already present. So if the line were to read:

```
ffd .dat sticky
```

the Data file **output** would be renamed **output.dat**, whereas the Data file **output.dat** would not be renamed.

When returning from UNIX the extension doesn't get removed; otherwise it's handled the same as ever, so the file type gets set using this extension.

We expect you'll want the 'To extensions:' part to duplicate the entries in the 'From extensions' part, so any extension that gets added when a file is transferred to UNIX gets removed again if the file returns to RISC OS. However, there may be a lot of UNIX extensions that you wish to convert to a single RISC OS file type. For example, you may have several UNIX applications each of which generates text files with different extensions – say '**.txt**', '**.doc**' and '**-asc**'. To do so, just add extra entries to the 'To extensions', thus:

```
Text .txt  
Text .doc  
Text -asc
```

- 7 When you've added all the extensions you want to, save the edited **extensions** file, overwriting the old one.



# 12 NFS star commands

## \*Free

Displays free space on a mount.

**Syntax**      **\*Free** [*mountname*]

**Parameters**    *mountname*                      The name of an NFS mount point

**Use**              \*Free displays your available free space on a mount, the total free space on that mount, and the size of the mounted filesystem. If the mounted filesystem is using quotas, \*Free shows your free and used space on it, the quota beneath which you are expected to remain, and the absolute limit of usage that you cannot exceed.

If no mountname is given, the current one is used.

**Example**            **\*Free tplusr**  
No quota on tplusr  
Space on filing system:  
Free 69 889K  
Available 33 168K  
Size 367 218K

```
*Free home
Bytes free      9 626K
Bytes used      374K
Bytes quota    10 000K
Bytes limit     12 000K
```

**Related  
commands**          None

## \*Logon

Sets the name server to use and/or authenticates a user/password pair.

**Syntax**      **\*Logon** [-Host *hostname*]  
                  [*username* [CR] *password* ]

|                   |                 |  |
|-------------------|-----------------|--|
| <b>Parameters</b> | <i>hostname</i> | The host to be used as the name server for this and subsequent name requests |
|                   | <i>username</i> | The user whose details are to be authenticated on the name server            |
|                   | <i>password</i> | The user's password, which – if omitted – you will be prompted for           |

**Use**            \*Logon sets the name server to use and/or authenticates a user/password pair, depending on the parameters passed:

- if a *hostname* is given, then it sets the name server to use for this and subsequent authentication using \*Logon
- if a *username* and *password* are given, they are passed to the current name server for authentication; they are also used to access any mount points you create in the future
- if no parameters are given, then the current user and name server are displayed.



## Examples

**\*Logon -Host tp1**

Sets tp1 as the current name server.

**\*Logon mhardy**

Prompts for the password, which is reflected as dashes; authenticates the pair using the current name server (ie tp1); if the password is valid, sets the current user to mhardy.

**\*Logon**

Displays the current user (mhardy) and the current name server (tp1).

## Related commands

\*Mount (page 108)

# \*Mount

Lists or mounts NFS mount points.

**Syntax**      `*Mount [-Host hostname] [mountname [mountpath]]`

|                   |                  |   |
|-------------------|------------------|---|
| <b>Parameters</b> | <i>hostname</i>  | The host to be used as the NFS server for this and subsequent mount requests      |
|                   | <i>mountname</i> | The name by which RISC OS will refer to this mount point                          |
|                   | <i>mountpath</i> | The directory on the NFS server which shall act as \$ on<br><b>NFS::mountname</b> |

**Use**      \*Mount lists or mounts NFS mount points, depending on the parameters passed:

- if a *hostname* is given, then it sets the host as the NFS server to be used for this and any subsequent mount requests; it also lists all mount points on the host, giving the mount name, mount path and user name for each mount point
- if a *mountname* and *mountpath* are given, then the *mountpath* is mounted, using the last user name that was successfully authenticated with \*Logon, and the URD, CSD, PSD and Library are all set to \$ (i.e. the *mountpath*)
- if no *mountpath* is given, but a *mountname* is given for which a mount already exists, then the URD, CSD, PSD and Library are all reset to \$ for that mount
- if no parameters are given, then all current mount points are listed, showing the mount name, mount path and user name for each mount point.

The files accessed via a mount point are always accessed using the details of the user which was current at the time the \*Mount command was executed.



## Examples

**\*Mount -Host tp1 tp1usr /usr**

Mounts the directory /usr on the host tp1, giving it the mount name tp1usr.

**\*Mount tp1usr**

Resets the URD, CSD, PSD and Library for the mount named tp1usr to \$ on that mount.

**\*Mount -Host tp1**

Lists all mount points on tp1.

**\*Mount**

Lists all mount points.

## Related commands

**\*Logon (page 106)**

## **\*NFS**

Selects NFS as the current filing system.

### **Syntax**

**\*NFS**

### **Use**

\*NFS selects NFS as the filing system for subsequent operations. Remember that it is not necessary to switch filing systems if you use the full pathnames of objects. For example, you can refer to ADFS objects when NFS is the current filing system.

### **Example**

**\*NFS**

### **Related commands**

\*ADFS

\*Net

\*RAM

\*ResourceFS

\*SCSI

\*LanMan

\*ShareFS



Displays NFS module internal statistics.

**Syntax**            **\*NFSInfo**

**Use**                \*NFSInfo displays detailed information about NFS module activity, including known hosts, users, URDs, CSDs, PSDs and Libraries, and details of its cache performance.

Most of the information displayed is difficult to interpret for non-experts. It is presented mainly as an aid to trouble-shooting, should you require it, perhaps to help when making a technical support query.

**Example**

```
*NFSInfo
Host list:
name=<tp1>, usage = 1
name=<tp2>, usage = 1
name=<tp3>, usage = 2

Cache hits 90051; Cache misses 36385

User list:
name=<mhardy>, uid=<1234>, gid=<27>, usage = 1
name=<nobody>, uid=<32767>, gid=<9999>, usage = 1
CSD: 0x1844144, <>
PSD: 0x1844144, <>
URD: 0x1844144, <>
Library: 0x1844144, <>
```

**Related  
commands**            None

## **\*TimeOffset**

Sets the time offset in minutes from GMT to local time.

**Syntax**      **\*TimeOffset** *minutes*

**Parameters**    *minutes*                      Minutes offset from GMT to local time

**Use**                \*TimeOffset sets the time offset in minutes from GMT to local time. RISC OS NFS adds this offset when it interprets the server's date stamps, which it assumes to be in GMT. The offset can be negative or positive.

**Example**          **\*TimeOffset** 60

**Related commands**    None



# 13 LanMan star commands

## \*Connect

Sets up a connection to a file server.

**Syntax**      *\*Connect name server share-name [username password]*

|                   |                   |  |
|-------------------|-------------------|--|
| <b>Parameters</b> | <i>name</i>       | Mount name   |
|                   | <i>server</i>     | File server name   |
|                   | <i>share-name</i> | Directory path   |
|                   | <i>username</i>   | The user whose details are to be authenticated on the server |
|                   | <i>password</i>   | The user's password  |

**Use**      \*Connect sets up a connection to a file server. It is functionally synonymous with \*Mount in other filing systems.

**Example**      *\*Connect Home OmniClient1 fbloggs  
Fredbloggs Atlas*

**Related commands**      \*Disconnect (page 114)

## \*Disconnect

Disconnects from a file server.

**Syntax**      \*Disconnect *name*

| Parameters | <i>name</i> | Mount name |
|------------|-------------|------------|
|------------|-------------|------------|

**Use** \*Disconnect disconnects from a file server.

**Example**      \*Disconnect Home

**Related commands**      \*Connect (page 113)



## **\*LanMan**

Selects Lan Manager as the current filing system.

**Syntax**            **\*LanMan**

**Use**                \*LanMan selects Lan Manager as the current filing system.

**Examples**        **\*LanMan**

**Related  
commands**        None

## **\*LMinfo**

Displays debugging information.

### **Syntax**

**\*LMinfo**

### **Use**

\*LMinfo displays information on current mounts, and debugging information including details of previous errors.

### **Example**

**\*LMinfo**

### **Related commands**

None



## **\*LMLogoff**

Logs a user off a workgroup mount.

**Syntax**            **\*LMLogoff**

**Use**                \*LMLogoff logs a user off a workgroup mount.

**Example**           **\*LMLogoff**

**Related  
commands**        \*LMLogon (page 118)

# \*LMLogon

Enables network browsing for LanManFS servers.

**Syntax**      *\*LMLogon workgroup username password*

|                   |                  |  |
|-------------------|------------------|--|
| <b>Parameters</b> | <i>workgroup</i> | Name of workgroup or domain name                             |
|                   | <i>username</i>  | The user whose details are to be authenticated on the server |
|                   | <i>password</i>  | The user's password  |

**Use**      \*LMLogon logs the username on as part of a workgroup. An LMLogon command must have been issued, usually from the !Omni.Files.Startup file, before automatic browsing of Lan Manager servers can take place.

**Examples**      *\*LMLogon domain guest guest*

**Related commands**      \*LMLogoff (page 117)



# **\*LMNameMode**

Sets the way LanManFS capitalises file and directory names

**Syntax**            **\*LMNameMode 0|1|2**

**Use**                \*LMNameMode sets the way LanManFS capitalises file and directory names:

- 0       Sets all capitals
- 1       Sets lower case
- 2       Preserves case (including a mixture of cases)

**Example**           **\*LMNameMode 0**

**Related  
commands**        None

# **\*LMprinters**

Adds a known server and list of printers

**Syntax**      **\*LMprinters server printer\_name(s)**

**Parameters**    *server*                      File server name  
                  *printer\_name*            Printer name

**Use**            \*LMprinters adds a known server and list of printers. It allows printers to be associated with servers manually: printers are usually automatically detected if server browsing is turned on, but if the **-n** parameter has been used with the \*LanMan command they will not be.

**Example**       **\*LMprinters omniclient printer\_green**

**Related  
commands**      None



## **\*LMserver**

Adds a known server and list of mount paths

**Syntax**      **\*LMserver** *server mount\_path(s)*

|                   |                   |                  |
|-------------------|-------------------|------------------|
| <b>Parameters</b> | <i>server</i>     | File server name |
|                   | <i>mount_path</i> | Mount path name  |

**Use**      \*LMserver adds a known server and list of mount paths to the list of network servers.

**Examples**      **\*LMserver** omni2 public

**Related commands**      None





# Index

## Symbols

- !BootNet 42
- !Configure file 53–54, 58–59
- !Internet 39
  - absolute programs 65
  - configuring 63
  - hosts file 63
- !Omni 13
- !Printers 22
- \* Commands
  - !Internet 64
  - Ethernet drivers 66
- \*ARP 60–61, 68–69
- \*Configure Ether3 70
- \*Connect 113
- \*Disconnect 114
- \*EnInfo 61, 71
- \*Free 105
- \*IfConfig 73–75
- \*IfRConfig 76
- \*InetGateway 77
- \*InetInfo 78
- \*LanMan 115
- \*LMinfo 116
- \*LMLogoff 117
- \*LMLogon 118
- \*LMNameMode 119
- \*LMprinters 120
- \*LMserver 121
- \*Logon 106
- \*Mount 108–109
- \*NFS 110
- \*NFSInfo 111
- \*Ping 79–80
- \*Route 55–56, 60, 81–82
- \*RouteD 54, 55–56, 59, 60, 83–86
- \*RouteDTraceOff 87
- \*RouteDTraceOn 88
- \*Status Ether3 89
- \*TimeOffset 101, 112

## A

- Absolute programs 65
- Acorn Access 10, 39
- Acorn Access+ 10, 39
- Address Resolution Protocol see ARP
- Alias 10
- Aliases
  - mount 21
- AppleTalk 10
- ARP 68–69, 74, 76
- ARP servers 55, 60
- AUN Level 4 10, 42
- Authenticator 19
- Auto-location 32, 118
- Automatic log on 20
- Automatic start up 14

## B

- Broadcast address 75
- Browsing 118

## C

- Client 10
- Client-name 91
- Command line
  - !Internet 64
- Configuration
  - printers 22
- Configuration options
  - startup file 31
- Configuring
  - !Internet 63
  - hosts file 63
- Configuring OmniClient 29
- Customising OmniClient 14

## D

- Databases 53, 58, 59
- Date stamps see file mapping (date stamps)
- DDN Network Information Center 50
- Desktop boot file 29

- Desktop boot file 14
- Dismounting 26
- Displaying available servers 17
- Distribution disc 56
- Driver 53, 57, 58
  - see also* Ethernet driver modules

- Drivers subdirectory 57

## E

- ECHO\_... datagrams 79
- Econet 50
  - file servers 57
- Editing
  - mounts file 34
- Encoding illegal characters 23
- Ethernet 45, 47, 50
  - interfaces 47
- Ethernet driver modules 66
  - information 71
- Ethernet drivers 66
- Ethernet 3 interfaces
  - configuration 70, 89

- Expert mode 32
  - dismounting 26
  - quitting OmniClient 27
  - short cuts 25

- Extension 91
  - file format 92
  - file mapping 91

## F

- File format
  - extensions 92
- File mapping 91–104
  - access attributes 97–98, 100
  - date stamps 98, 101
  - extensions 91, 96, 99, 101
  - extensions file 103–104
  - file contents 96, 100
  - file types 101
  - filenames 95, 99–100
  - finding an object 98
  - flags 92, 94
  - soft links 102
  - untyped files 96, 101

- File printing 22
- Filer 10
- Filer functions 26
- Files

- dead 96
  - extensions 29
  - mounts 29
  - StartShare 29
  - startup 29

- Files subdirectory 59–60

- Filetypes 91, 92

- Filing system 10
  - current 110

- Finding an object *see* file mapping (finding an object)

- Flags 91

- file mapping 92, 94
  - in the mount file 36
  - locked 31

- Free space 26

## G

- Gateways 54, 55–56, 59, 60, 85
  - database 85

## H

- Hidden flag 36
- Host addresses *see* Internet addresses
- host names 48, 53–54, 58, 59
- Hosts file 54, 59, 63
  - see also* databases

## I

- IBM OS/2 10, 40

- Icon bar
  - mounting servers from 19
  - actions 25

- Icons
  - server 18

- Illegal characters
  - encoding 23

- Inet\$EcoIPAddr 58

- Inet\$EcoIPMask 58

- Inet\$EtherDevice 58

- Inet\$EtherIPMask 58



- Inet\$HostName 58
- Inet\$IsGateway 59
- Inet\$RouteOptions 59, 84
- Inet\$Startup 58, 60
- InetDBase\$Path 58, 59
- Information
  - about a mount 26
- Installing OmniClient 13
- interface
  - name 73, 76
  - unit number 73, 76
- Interface names 48, 54
- Internet addresses 49–51, 53–54, 58, 59, 76
  - classes 50
- Internet application
  - starting 64
- Internet module 65
  - \* Commands 64
  - information 78
- Internet service providers 50
- IP address
  - setting 63
- L**
- Lan Manager 10, 40
- LanManFS 32, 40
- Level 4 42
- Licence agreement 5
- Loading protocol modules 32
- Locked flag 31, 36
- lpr 24
- M**
- MAC addresses *see* physical addresses
- Mapping 91
- Microsoft Windows 9
- Modules
  - Ethernet drivers 66
  - loading 32
- Mount 10
- Mount aliases 21
- Mount point 10
- Mounting
  - additional servers 21
  - automatically 20
  - file server 18
  - from File servers window 19
  - from icon bar 19
- Mounts
  - dismounting 26
  - finding free space 26
  - information 26
- Mounts file 33
  - editing 34
  - flags 36
  - removing mounts 35
  - saving 33
  - supporting multiple 30
- Mouse buttons 11
- N**
- Netmasks 49–50, 53, 58, 74, 76
- Network addresses *see* Internet addresses
- Network printing 21
- Network servers window 17
- Networks database 74
- Networks file *see* databases
- NFS 10, 19, 41
  - printing with 23
- NFS module
  - information 111
- NFS mounts
  - free space 105
  - listing 108
  - mounting 108
  - name server 106
- NFS\$CreateAccess 97
- NFS\$TruncateLength 99
- Novell Netware 10
- NT Advanced Server 10
- O**
- Omni\$Options 31
- Omni\$User 32
- OmniClient
  - configuring 29
  - installing 13
  - quitting 27
- OmniNFS 41
- OS/2 Warp 10, 40



## P

- Packet forwarding 54, 59, 77
- Packet loss statistics 80
- Packing list 7
- pcnfsd 24
- physical addresses 51, 55, 61, 68–69, 71
- Ping 79
- Preset flag 36
- Principal host names *see* host names
- Printing 21
  - configuration 22
  - to file 22
  - using NFS 23
- Protocol 11
- Protocols 39
  - loading modules 32
- Protocols file *see* databases

## Q

- Quitting OmniClient 27

## R

- Removing
  - mounts 35
- Research Machines NetLM 10
- Reverse ARP 55, 58, 60
- RIP *see* Routing Information Protocol
- RISC OS 9
  - character encoding 23
- Root directory
  - opening 26
- RouteD 66
- Routing 55–56
- Routing Information Protocol 56
- Routing tables 81, 83–86
- Run\$Path 65

## S

- Saving
  - mounts file 33
- Security 31
- Server 11
- Servers
  - displaying list 17
  - icons 18

mounting 18

## Setting

- IP addresses 63
- Setting user name 32
- Share 11
- Shortcuts 25
- Startup file 31
- startup file 53, 58, 60
- subnet addresses *see* Internet addresses
- Supporting multiple mounts files 30

## T

- TCP/IP networks 48
- TCP/IP Protocol Suite 10
- Timezones 112

## U

- UNIX 41
- Unix 10
- User name
  - setting 32
- Using OmniClient 17

## W

- Window
  - network servers 17
- Windows 9
- Windows for Workgroups 10, 40
  - and auto-location of servers 32
- Windows NT 40







ANT Limited

PO Box 300

Cambridge

CB1 2EG

Telephone: 01223 567808

Fax: 01223 567801

Email: [support@ant.co.uk](mailto:support@ant.co.uk)